Quick BASIC
プログラミング500題

田中　廣――著

日刊工業新聞社

Quick BASIC
プログラミング500題

田中 廣――著

日刊工業新聞社

Quick BASIC
プログラミング500題

田中　廣──著

日刊工業新聞社

はじめに

“コンピュータ言語として何がよいか”という議論があります．とくに“パソコン用として何がよいか”となるとユーザー層がひろいだけに，一層多くの議論があります．最近では，パソコン用言語としては，“BASIC”対“C 言語”の対比という形で一般化されています．そのおのおのの長所，欠点はいろいろなところでとりあげられていますので省略しますが，一言でいえば“BASIC”のフレンドリー（親しみやすい，わかりやすい）な面はすばらしいと思います．

“Quick BASIC”はフレンドリーを残しながら，従来の BASIC の欠点といわれてきた，①構造化，②モジュール化を可能にしました．インタープリタ形式のように，プログラムの 1 行ごとの入力チェックを行いながら，全体をコンパイルするコンパイル形にしています．コンパイル形の場合，コンパイル，リンクと作業が複雑になるのですが，“統合環境”と呼ぶエデット（プログラムの作成）→コンパイル（オブジェクトファイルの作成）→リンク（実行可能ファイルの作成）を簡単に行える環境を準備しています．加えて，デバッガによるプログラムのチェックも簡単にできるようになっています．つまり，“BASIC”のよさを残しながら“C 言語”のよさをとりこんでいるわけです．したがって，入門者から一般のユーザーあるいはプログラム開発の専門家まで，“BASIC”をもう一度見直し，十分に使っていけるような言語になっています．

本書は「I　準備編」で，“Quick BASIC”のバックアップコピーのとり方，セットアップの仕方から，ドロップダウンメニューの操作・機能を説明しています．II編以降と並行して使ってください．「II　プログラム編」，「III　グラフィック編」はおのおの“Quick BASIC”を使ったプログラムのつくり方です．「IV　応用編」では従来の N-BASIC との関連，再帰プログラム等を紹介しています．例題 272 題，演習問題 211 題，応用問題 20 題による，“プログラミング 500 題”シリーズの一環としています．

なお，“Quick BASIC” Ver. 4. 5（音楽等の機能が追加されています．II編 26 章参照）が最新版ですが Ver. 4. 2 でも使えるように工夫してあります．

確認用に使用した機器は PC-9801 VM です．PC-9800 シリーズで使えますが，一部の機種のハイレゾリューションモード（XA，LT，XL，XL 2，RL）では使えません．OS は MS-DOS を使っています．MS-DOS は現在 Ver. 3. 3 が最新ですが 2. 0 以上で使用できます．

1989 年 11 月

田中　　廣

目　　次

I　準備・環境コマンド編

II　プログラム編

III　グラフィック編

IV　応用編

Ver.4.5のメニュー画面

GRPH キーとメニューキー(F , E , V , S , R , D , C , O , H)をおすことでメニューが選択されます．サブメニューは先頭の1文字をおすか，カーソルを矢印キーを使って動かし ↵ します．

F/ファイル

N/新規
O/読込...
M/結合...
S/保存
A/名前を変えて保存...
V/すべて保存

C/サブファイル作成...
L/サブファイル読込...
U/サブファイル解放...

P/印刷...
D/MS-DOSコマンド

X/終了

E/編　集

U/元に戻す	GRPH+BS
T/カット	SHIFT+DEL
C/コピー	CTRL+INS
P/ペースト	SHIFT+INS
E/削除	DEL
S/新規SUB...	
F/新規FUNCTION...	

V/表　示

S/SUB 一覧	F2
E/次のSUB	SHIFT+F2
P/画面分割	
N/次のステートメント	
U/実行画面	F4
I/インクルードファイル編集	
L/インクルードファイル表示	

S/検　索

F/検索...	
S/指定文字列検索	CTRL+¥
R/次を検索	F3
C/置換...	
L/ラベル検索...	

R/実　行

S/スタート	SHIFT+F5
R/リスタート	
N/継続	F5
C/COMMAND$の編集...	
X/実行ファイル作成...	
L/ライブラリ作成...	
M/メインモジュールの設定...	

D/デバッグ

A/ウォッチの追加...
I/簡易ウォッチ...　　SHIFT+F9
W/ウォッチポイント...
D/ウォッチの削除...
L/全ウォッチの削除

T/トレース
H/ヒストリ

B/ブレークポイント　　F9
C/全ブレークポイントの削除
E/エラー時強制中断
S/次のステートメントの設定

C/関　数

O/オプション

D/画面設定...
S/サーチパス設定...
R/マウス機能選択...
*Y/構文チェック
*F/Fullメニュー

H/ヘルプ

I/索引
C/目次
T/キーワード：　　F1
H/ヘルプの利用法　　SHIFT+F1

（注）
Ver. 4.5ではイージーメニューでたちあがりますが，GRPH，O でオプションメニューを選び，F をおしてフルメニューをセットします．4.2 とほぼ同じメニュー構成です．

（注）Ver.4.2 と Ver.4.5 の違い．

Ver. サブメニュー	Ver.4.5	Ver.4.2
Y／構文チェック	O／オプション	E／編集
O／オプション	O／オプション	V／表示
I／簡易ウォッチ	D／デバッグ	な　し
E／エラー時 強制中断	D／デバッグ	な　し

言葉の違いは省略

Ver.4.2のメニュー画面

GRPH キーとメニューキー(F , E , V , S , R , D , C , H)をおすことでメニューが選択されます. サブメニューは先頭の1文字をおすか, カーソルを矢印キーを使って動かし ↵ します.

F/ファイル

N/新規
O/読込...
M/追加...
S/保存
A/名前を変えて保存...
V/すべて保存

C/サブファイル作成...
L/サブファイル常駐...
U/サブファイル解放...

P/印刷...

D/MS-DOSコマンド

X/終了

E/編　集

U/元に戻す	GRPH+BS
T/カット	SHIFT+DEL
C/コピー	CTRL+INS
P/ペースト	SHIFT+INS
E/削除	DEL

S/新規SUB...
F/新規FUNCTION...

Y/構文チェック

V/表　示

S/SUB 一覧...	f･2
E/次のSUB	SHIFT+f･2
P/画面分割	

N/次のステートメント	
U/実行画面	f･4

I/インクルードファイル編集
L/インクルードファイル表示

O/オプション

S/検　索

F/検索...	
S/指定文字列検索	CTRL+¥
R/次を検索	f･3
C/置換...	
L/ラベル検索...	

R/実　行

S/スタート	SHIFT+f･5
R/リスタート	
N/続行	f･5
C/COMMANDS$ 入力...	

X/EXE ファイル作成...
L/ライブラリ作成...

M/メインモジュールの設定...

D/デバッグ

A/ウォッチの追加...
W/ウォッチポイント...
D/ウォッチの削除...
L/全ウォッチの削除

T/トレース
H/ヒストリ

B/ブレークポイント f･9
C/全ブレークポイントの解放
S/カレントステートメントの設定

C/関　数

H/ヘルプ(f･1)

I 準備・環境コマンド編

1章　準　備

▶セットアップまでに必要なディスク

Ver.4.5（最低8枚）　　Ver.4.2（最低4枚）

〔オリジナル〕　4枚　セットアップ　コンパイラ　ライブラリ　アドバイザ

〔オリジナル〕　2枚　プログラム　ライブラリ&サンプル

ディスクコピー

〔新しいディスク〕バックアップ　4枚　（フォーマットして使う）

2枚　（フォーマットして使う）

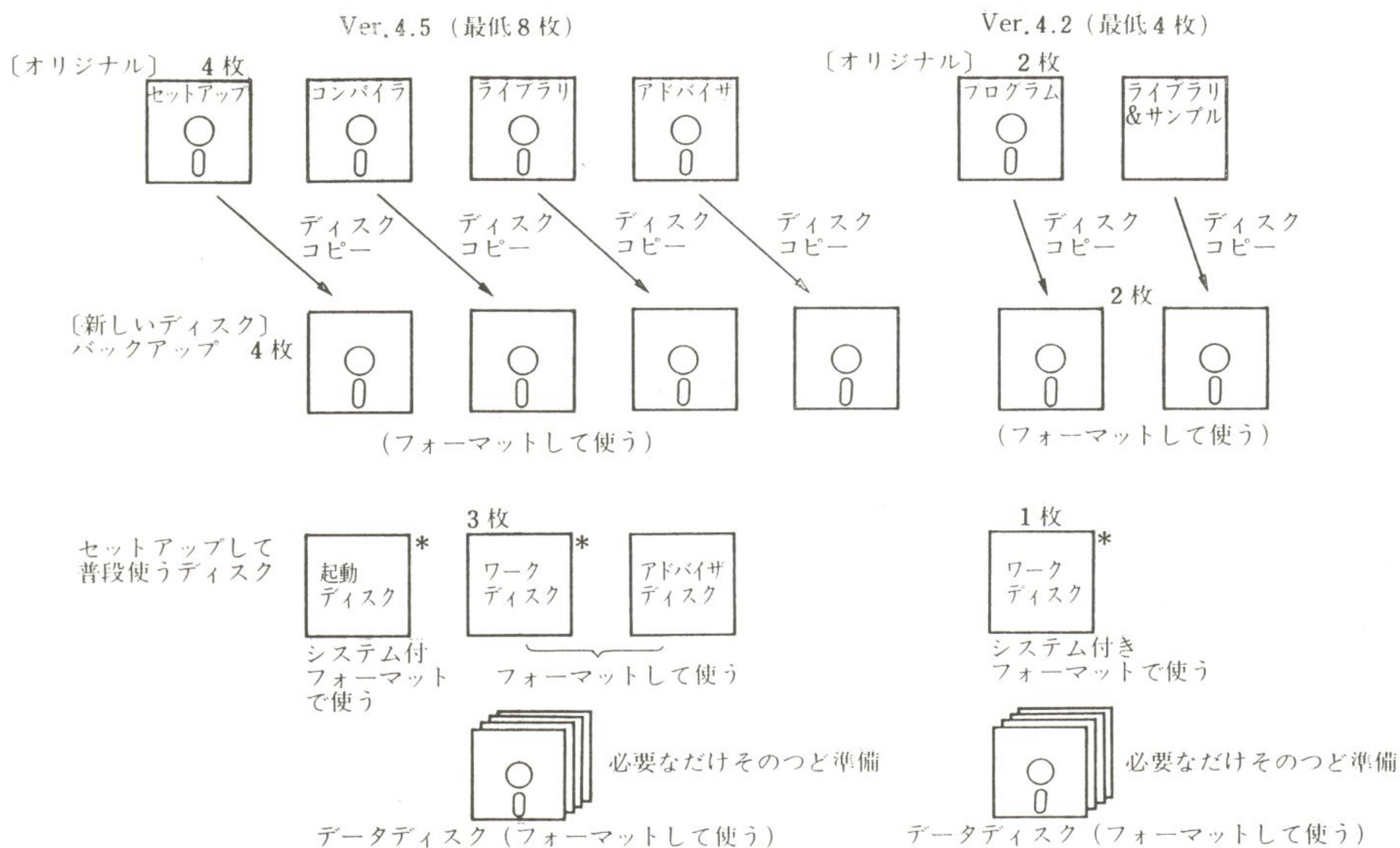
セットアップして普段使うディスク

3枚　起動ディスク*　ワークディスク*　アドバイザディスク

システム付フォーマットで使う　　フォーマットして使う

1枚　ワークディスク*　システム付きフォーマットで使う

必要なだけそのつど準備　データディスク（フォーマットして使う）

必要なだけそのつど準備　データディスク（フォーマットして使う）

＊Quick BASIC を使うときのメインのディスクを，Ver.4.5では『起動ディスク』と呼び，Ver.4.2では『ワークディスク』と呼んでいる．Ver.4.5では『ワークディスク』の名の別のディスクがあるので，呼び方に注意して下さい．

【例題1】　バックアップ（メニュー方式によるフォーマットとディスクコピー）

Quick BASIC の4枚(バージョン4.2以下では2枚)のディスクをバックアップのためコピーせよ．

【解　説】　◎　ディスクのバックアップをとります．

① MD-DOS を使用してバックアップをつくります．手順としては，新しいディスクのフォーマット

とコピーです．MS-DOS のバージョンによって若干手続きが異なりますが，本書ではバージョン 3.30を使って行います．

② MS-DOS のディスクを A ドライブに入れ，コンピュータの電源を入れます．次のメニュー画面と異なります．

```
MS-DOS　コマンド　メニュー　　　　（コマンド選択） 1/ 6　　　Menu v2.20
F1 README．DOCファイルの表示　　　　　　　　　　　日付：
F2 アプリケーションの登録　　　　　　　　　　　　　　1989-08-20
F3 フロッピーディスクの初期化　　　　　　　　　　　時刻：
F4 フロッピーディスクのバックアップ　　　　　　　　　01:28
F5 固定ディスクの初期化　　　　　　　　　　　　　　MS-DOS:
F6 ディスク内全ファイルのコピー　　　　　　　　　　　Ver. 3.30
F7 システムファイルの転送
F8 ディレクトリ情報の表示
F9 MENUの終了
A>TYPE README.DOC
MS－DOSを使用するにあたっての注意事項を表示します．
矢印キーで項目を選択し、リターンキーを押してください
```

③ [↓] キーをおして 2 頁目の F8 にカーソルをおきます．

```
MS-DOS　コマンド　メニュー　　　　（コマンド選択） 2/ 6　　　Menu v2.20
F1 フロッピーディスク（640KBﾀｲﾌﾟ）の初期化　　　　日付：
F2 フロッピーディスク（640KBﾀｲﾌﾟ）のバックアップ　　1989-08-20
F3 ディスク内全ファイルのコピー（ドライブ指定）　　時刻：
F4 ファイル内容の表示　　　　　　　　　　　　　　　　01:34
F5 日付の表示と設定　　　　　　　　　　　　　　　　MS-DOS:
F6 時刻の表示と設定　　　　　　　　　　　　　　　　　Ver. 3.30
F7 ディスクの初期化（メニュー形式）
F8 ディスクのコピー／照合（メニュー形式）
F9 フロッピーディスクのバックアップ（ドライブ指定）
A>FORMAT %: & DISKCOPY %: %:
フロッピーディスクの複写を作成します．（ドライブの指定ができます）
矢印キーで項目を選択し、リターンキーを押してください
```

④ [↵] によって次の画面となります．まずフォーマットです．"A:"の部分が反転していますので [↓] で"B:"へ移します．

```
ＭＳ－ＤＯＳ　コマンド　メニュー　　　（ドライブ選択）　1/ 1　　　Menu v2.20

      A:
      B:
      C:
      D:
      E:

A>FORMAT B: & DISKCOPY %: %:

矢印キーでドライブを選択し、リターンキーを押してください
```

⑤ ⏎ により次の画面となります．次はコピーの元のドライブとコピー先のドライブの指定です．まずコピー元の"A:"が反転しています．

```
ＭＳ－ＤＯＳ　コマンド　メニュー　　　（ドライブ選択）　1/ 1　　　Menu v2.20

      A:
      B:
      C:
      D:
      E:

A>FORMAT B: & DISKCOPY A: %:

矢印キーでドライブを選択し、リターンキーを押してください
```

⑥ ⏎ により次の画面になります．次はコピー先ですから"B:"へ反転を移します．

```
MS-DOS コマンド メニュー          (ドライブ選択) 1/ 1        Menu v2.20

     A:
     B:
     C:
     D:
     E:

A>FORMAT B: & DISKCOPY A: B:

矢印キーでドライブを選択し、リターンキーを押してください
```

⑦ ↵ します．次の画面へ移ります．まずフォーマットをします．

```
A>FORMAT B:
Format Version 4.10

新しいディスクをドライブ B: に挿入し
どれかキーを押してください
```

⑧ 上の表示に従い，新しいディスクをドライブBに入れ ↵ します．次に，ディスクのタイプを問合わせます．2HD の 1MB タイプを使用するときは 2 ↵ です．

```
A>FORMAT B:
Format Version 4.10

新しいディスクをドライブ B: に挿入し
どれかキーを押してください

ディスクのタイプは  1 : 640(KB)  2 : 1(MB)  =  2
```

⑨ フォーマットが始まります．終了すると次の問合わせに入ります．

```
A>FORMAT B:
Format Version 4.10

新しいディスクをドライブ B: に挿入し
どれかキーを押してください

ディスクのタイプは  1 : 640(KB)  2 : 1(MB)  = 2

目的のディスクは 1MB FD です

フォーマットが終了しました
0  10  20  30  40  50  60  70  80  90  100
                                          (%)

   1250304 バイト  全ディスク容量
   1250304 バイト  使用可能ディスク容量

別のディスクをフォーマットしますか(Y/N)
```

フォーマット終了時の表示

終了の問合わせ

⑩ 今回は4枚フォーマットします．バージョン4.2以下では2枚です．したがって，Y とします．

⑪ Bドライブのディスクをとり，新しいディスクを挿入し ↵ します．同じようにフォーマットがすむと次のフォーマットの問合わせとなります．

```
別のディスクをフォーマットしますか(Y/N) y


新しいディスクをドライブ B: に挿入し
どれかキーを押してください

目的のディスクは 1MB FD です


フォーマットが終了しました
0  10  20  30  40  50  60  70  80  90  100
                                          (%)

   1250304 バイト  全ディスク容量
   1250304 バイト  使用可能ディスク容量


別のディスクをフォーマットしますか(Y/N)
```

⑫ Y をおして，Bドライブから2枚目のディスクをとり3枚目を入れ ↵ します．フォーマットが終わると次のフォーマットの問合わせとなります．

⑬ フォーマットを終了しますので N とします．

```
別のディスクをフォーマットしますか(Y/N) n
```

⑭ バックアップに入ります．バックアップはディスクのコピーです．

```
別のディスクをフォーマットしますか(Y/N) n
A> DISKCOPY A: B:

DISKCOPY version 4.00

ディスクのコピーを行います

送り側ディスクをドライブ A: に挿入してください
受け側ディスクをドライブ B: に挿入してください
準備ができたらどれかのキーを押してください
```

⑮ Aドライブの DOS のディスクをぬき，Quick BASIC の"セットアップ"(バージョン 4.2 以下では"プログラム")ディスクを入れます．B ドライブでは今フォーマットの終わったディスクが入っていますからそのまま ↵ します．

⑯ Aドライブのディスクが B ドライブのディスクにコピーされます．終了すると次の問合わせとなります．Y により次のコピーを行います．

```
A> DISKCOPY A: B:

DISKCOPY version 4.00

ディスクのコピーを行います

コピーは終了しました
もう一度実行しますか(Y/N)Y
```

⑰ ドライブ A を Quick BASIC の"コンパイラ"(バージョン 4.2 以下では"ライブラリ")ディスクに入れかえ，ドライブ B は新しいフォーマットずみのディスクに入れかえて ↵ します．バージョン 4.5 では同様に"ライブラリ"ディスク，"アドバイザ"ディスクのコピーをとります．バックアップが終ったらドライブ A を DOS のディスクと入れかえ，B ドライブはディスクをとり出します．

```
DISKCOPY version 4.00

ディスクのコピーを行います

コピーは終了しました
もう一度実行しますか(Y/N)N



コマンド選択に戻ります
準備ができたら、どれかキーを押してください
```

⑱ ↵ キーをおすと DOS のメニュー画面へもどります．

【例題 2】 ファイルの確認

バックアップされたディスクのファイルを確認せよ．

【解 説】 ◎ バックアップファイルの内容の確認をします．

① バックアップされたディスクのファイルを確認します．次のようになっていればコピーは完全です．

［Ver.4.5］

(セットアップ・ディスク)

```
ドライブ B: のディスクのボリュームラベルはありません．
ディレクトリは B:¥

PACKING  LST     13570  89-10-17  18:09
QBE      DOC      4475  89-09-06  18:33
README   DOC      4673  89-10-13  15:40
SAMPLE   DOC      2942  89-10-13  14:22
SETUP    EXE     61441  89-10-13  19:12
SETUP    @@@      3244  89-10-13  19:53
SAMPLE        <DIR>     89-10-18  16:27
LEARN         <DIR>     89-10-18  16:29
TRD07    TXT      8425  89-10-17  22:44
```

```
        9 個のファイルがあります.
   690176 バイトが使用可能です.

 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥

BIN          <DIR>      89-10-18   16:29
MOUSE        <DIR>      89-10-18   16:30
        2 個のファイルがあります.
   107520 バイトが使用可能です.
```

(コンパイラ・ディスク)

```
 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥

LIB          <DIR>      89-10-18   16:32
USERLIB      <DIR>      89-10-18   16:33
        2 個のファイルがあります.
   545792 バイトが使用可能です.
```

(ライブラリ・ディスク)

```
 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥

QBADVR       <DIR>      89-10-18   16:35
        1 個のファイルがあります.
   591872 バイトが使用可能です.
```

(アドバイザ・ディスク)

〔**Ver.4.2**〕

```
QB_ENV          <DIR>       88-12-15     4:20
QB_CMP          <DIR>       88-12-15     4:20
LIB_E           <DIR>       88-12-15     4:20
SETUP     DOC         2763  88-12-15     4:20
SETUP     EXE        30198  88-12-15     4:20
PACKING   LST         3303  89-01-15     4:20
README    DOC         7832  89-01-15     4:20
SAMPLE    DOC         4196  89-01-15     4:20
        8 個のファイルがあります.
   176128 バイトが使用可能です.
```

(プログラムディスク)

```
LIB_A           <DIR>       88-12-15     4:20
SOURCE          <DIR>       88-12-15     4:20
        2 個のファイルがあります.
   818176 バイトが使用可能です.
```

(ライブラリディスク)

② 確認は次のようにします．MS-DOS 3.30 の場合はメニュー画面から"F8"を選びます．

```
MS－DOS　コマンド　メニュー　　　　　　（コマンド選択） 1/ 6　　　Menu v2.20
F1 README．DOCファイルの表示                                  日付：
F2 アプリケーションの登録                                        1989-08-20
F3 フロッピーディスクの初期化                                  時刻：
F4 フロッピーディスクのバックアップ                               03:20
F5 固定ディスクの初期化                                        MS-DOS:
F6 ディスク内全ファイルのコピー                                   Ver. 3.30
F7 システムファイルの転送
F8 ディレクトリ情報の表示
F9 MENUの終了
A>DIR %: /P
ディレクトリの内容を表示します．
矢印キーで項目を選択し、リターンキーを押してください
```

③ ⏎ をします．次の画面で"B:"を選び，ディスクをBドライブに入れ ⏎ します．"A:"を選んでディスクをAドライブに入れて ⏎ しても同じです．

```
MS－DOS　コマンド　メニュー　　　　　　（ドライブ選択） 1/ 1　　　Menu v2.20
    A:
    B:
    C:
    D:
    E:
A>DIR B: /P
矢印キーでドライブを選択し、リターンキーを押してください
```

④ メニューを使わない場合はBドライブにディスクを入れて

DIR B: ⏎

あるいはAドライブにディスクを入れて

DIR A: ⏎ です．

【例題3】　フォーマット（ふだん使う Quick BASIC 用ディスク）
新しいディスクをフォーマットせよ.

【解　説】　◎　フォーマットをします.

① 新しいディスクのフォーマットをします.

② A ドライブに MS-DOS のディスクを入れ電源を入れます. メニュー方式によるフォーマット例は【例題 1】のとおりです. ここではメニュー方式によらない例を示します. メニューから [f9] をおします.

A>

の表示となります.

③ 次のようにかき [↵] します.

A>FORMAT B:　[↵]

④ 【例題 1】の項⑦となりますので, B ドライブに新しいディスクを入れ [↵] します. 続いて 1MB の指定のため [2] [↵] とします.

⑤ フォーマットが終ると次のフォーマットをするかどうかを問合わせます.

別のディスクをフォーマットしますか(Y/N) y

⑥ [Y] をおすと同じ手順で, 新しいディスクを B ドライブに入れ [↵], [2] [↵] とします.

⑦ 3 枚フォーマットをして下さい. バージョン 4.5 では 2 枚は Quick BASIC のワークディスクとアドバイザディスク用, 1 枚は Quick BASIC のデータ用です. バージョン 4.2 以下ではデータ用のみ 1 枚です. フォーマットを終了したとき, 別のディスクをフォーマットしますか[Y/N]で [N] をおすと終了します.

【例題4】　バックアップ（非メニュー方式）
Quick BASIC の 4 枚（バージョン 4.2 以下では 2 枚）のディスクをバックアップコピーせよ.

【解　説】　◎　バックアップコピーをとります.

① メニュー方式を使わずに, バックアップ・コピーをとります.

② メニューから F9 をおし A> とします. A>から次のようにかきます.

A>DISKCOPY A:B:

A ドライブから B ドライブへコピーする命令です. [↵] します.

③ 次の表示になりますので A ドライブに Quick BASIC の"セットアップ"（バージョン 4.2 以下では"プログラム"）ディスクを入れ, B ドライブにフォーマットしたディスクを入れ [↵] します.

```
DISKCOPY version 4.00

ディスクのコピーを行います

送り側ディスクをドライブ A: に挿入してください
受け側ディスクをドライブ B: に挿入してください
準備ができたらどれかのキーを押してください
```

④ ディスクのコピーが終ると次の表示となり次のコピーをするかどうか問合わせます.

```
コピーは終了しました
もう一度実行しますか(Y/N)Y
```

⑤ [Y] をおすと項③へもどりますので A ドライブに Quick BASIC の"コンパイラ"（バージョン 4.2 以下では"ライブラリ"）ディスク，B ドライブにフォーマットした別の新しいディスクを入れて [↵] します.

コピーが終ると項④の問合わせとなります. バージョン 4.5 では同様に"ライブラリ"ディスク，"アドバイザ"ディスクをコピーします. コピーが終ったら次へすすみます.

⑥ A ドライブに DOS のディスクを入れ，B ドライブはコピーずみのディスクを抜いて [N] とします. バックアップは終了です.

【例題 5】　システム付フォーマット

新しいディスクをシステム付きでフォーマットせよ.

【解　説】 ◎ システム付フォーマットをします.

① Quick BASIC の起動ディスク（バージョン 4.2 以下ではワークディスク）をつくるディスクはシステム付きのディスクにセットアップする必要があります. そのためシステム付フォーマットを 1 枚行います.

② DOS のディスクを A ドライブに入れ電源を入れます. メニュー画面で [f9] をおし A>とし，A>で次のように記入します.

```
A>FORMAT B:/S
```

③ [↵] によって，【例題 1】と同じ手順でフォーマットが行われ，終了します.

```
Format Version 4.10

新しいディスクをドライブ B: に挿入し ―――――― 新しいディスクを B
どれかキーを押してください                     ドライブに入れ [↵]

ディスクのタイプは  1 : 640(KB)  2 : 1(MB)  = 2 ―――― [2] [↵]

目的のディスクは 1MB FD です

フォーマットが終了しました
0  10  20  30  40  50  60  70  80  90  100
                                        (%)
```

```
システムを転送しました

  1250304 バイト  全ディスク容量
   120832 バイト  システム領域
  1129472 バイト  使用可能ディスク容量

別のディスクをフォーマットしますか(Y/N)
```
N で終了

④ このディスクをDIRでみるとCOMMAND.COMというファイルが1つ入っています（正確には他に2つのかくれたファイルがコピーされています）.

```
A>dir b:

 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥

COMMAND   COM     24931  88-07-13    0:00
        1 個のファイルがあります.
  1129472 バイトが使用可能です.
```

これをシステム付きディスクといいます．このディスクを使うとDOSを立ちあがらすことができます.【例題1】でつくったディスクはシステムがありません．したがって，DIRでみるとファイルが入っていません.

```
A>dir b:

 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥

ファイルが見つかりません.
```

【例題6】　セットアップ

Quick BASICをセットアップせよ.

【解　説】 ◎ Quick BASICをセットアップします.

① DOSのディスクをAドライブに入れ，電源を入れて，DOSをたちあがらせます．日付け，日時をセット（または2回 ↵ キーをおす）すると次のAプロンプトとなります（メニュー方式の場合は f·9 をおす).

```
A>
```

② まずバージョン4.5でのセットアップを説明します．新しいシステム付きフォーマットすみのディスク1枚と，システムなしディスク2枚を用意します.

③ Aドライブにセットアップディスクを入れます．次のように入力します.

```
A>SETUP ↵
```

④ あとは画面の指示に従うだけですが，順を追ってみていきます.

まずセットアップユーティリティによりメニューが出ます.

```
QuickBASIC　セットアップ　ユーティリティ                    Version 1.00

        ********* < メニュー > ***********************************
          1. QuickBASICの利用環境を作成します。
          2. QBの概要、操作方法の説明を行います。
          3. セットアップを終了します。
        **********************************************************

          ●メニューを選択して下さい。・・・・・・・・・・・・?1

--------------------------------------------------------------------------
  QuickBASICの利用環境を作成します。
  始めてお使いになる方は必ず 1のメニューを選んでください。
  途中でセットアップを中断するには、CTRL+CもしくはSTOPキーを押して
  ください。この場合は、再度セットアップを起動して最初から行う必要があります。

  2のメニューを選択すると、QuickBASICの利用環境の作成は行わずに、
  QBの概要や操作方法を説明する「QBチュートリアル」が起動します。
  利用環境の設定を行う場合は1を選択してください。
```

1の"Quick BASICの利用環境を作成します"を選ぶため，"1"を入力します．[1] [↵] です．

⑤

```
QuickBASIC　セットアップ　ユーティリティ                    Version 1.00
  ●転送元のドライブ名を設定してください?A

--------------------------------------------------------------------------
  QuickBASICのマスターディスクの入っているドライブ名を設定してくだ
  さい。
```

オリジナルディスクの入るドライブを指定します．[A] [↵] とします．オリジナルディスクは常にAドライブに入れ，ふだん使うディスクはBドライブに入れてセットアップするものとします．

⑥

```
QuickBASIC セットアップ ユーティリティ                    Version 1.00

 ●転送元のドライブ名を設定してください？A
 ●どのドライブにインストールしますか？B

------------------------------------------------------------------
 QuickBASICをインストールするドライブ名を入力してください。
 フロッピーディスクにインストールする場合は、以下の準備が整っているしている
 ことを確認してください。
   1．フォーマット済みフロッピーディスクの準備。
      （DD版の時は4枚、HDの時は3枚のディスクが必要です。）
   2．パッケージに添付されている黒色の作業ディスクラベルを貼る。
   3．［QB起動ディスク］にMS－DOSを転送する。
 準備ができていない時は、CTRL＋Cでセットアップを終了し、ディスクを用意
 してから再度セットアップを起動してください。
```

セットアップするドライブを指定します．Bドライブとします．[B] [↵] です．

⑦

```
QuickBASIC セットアップ ユーティリティ                    Version 1.00

 ●転送元のドライブ名を設定してください？A
 ●どのドライブにインストールしますか？B

 ●上記の設定でよろしいですか[Y/N]？Y
------------------------------------------------------------------
 上記の設定でよければYを、変更したければNを指定してください。
 Nが指定された場合は、この画面の最初から指定しなおすことができます。
 この時、前回の設定がディフォルト値として表示されます。
```

設定の可否を確認します．Aドライブをオリジナル，Bドライブにセットアップするディスクを入れますので，[Y] [↵] とします．

⑧

```
QuickBASIC セットアップ ユーティリティ                    Version 1.00
 ●数値演算ライブラリを選択してください[1:Alt Math / 2:Emulator Math]？ 1

-----------------------------------------------------------------------
 QuickBASICは、2種類の数値演算ライブラリをサポートしています。
 ここでは、数値演算コプロセッサを装着した機種上で非常に高速な演算処理を保証
 するエミュレータライブラリ(Emulator Math)と，装着／非装着にかかわらずある程
 度のスピードで動作する代替演算ライブラリ(Alt Math)のどちらを使うかを選択し
 ます。
 Emulator 版の QuickBASIC インタープリタのファイル名は QBE.EXE です。プログ
 ラム名が異なることにご注意ください。なお、セットアップディスクの QBE.DOC に
 は、より詳しい情報が記載されています。Emulator 版をご使用の場合には必ずお読
 みいただくようお願いします。
```

数値演算ライブラリを問合わせます．通常は [1] [↵] です．

⑨

```
QuickBASIC セットアップ ユーティリティ                    Version 1.00
 ●数値演算ライブラリを選択してください[1:Alt Math / 2:Emulator Math]？ 1
 ●マウスを使用しますか[Y/N]？ Y

-----------------------------------------------------------------------
 マウスを利用することによって、統合環境の操作性がさらに向上します。ご利用を
 お勧めします。
 なお、マウスを利用する場合は、必ずマイクロソフトのマウスドライバ Version 6
 を利用してください。他のバージョンのマウスドライバでの利用はQuickBA
 SICの動作を保証できません。
```

マウスの使用または不使用を入力します．使用するときは [Y] [↵] です．

⑩

```
QuickBASIC　セットアップ　ユーティリティ                    Version 1.00

 ●数値演算ライブラリを選択してください[1:Alt Math / 2:Emulator Math]？1
 ●マウスを使用しますか[Y/N]？Y
 ●マウスの種類を選択してください[1:バスマウス / 2:シリアルマウス]1

----------------------------------------------------------------------
 ご使用のマウスの種類によって、1 もしくは 2 を選択してください。
 ＲＳ－２３２Ｃポートに接続されているマウスは 2:シリアルマウス、それ以外の
 ものは、1:バスマウスを選択してください。
```

マウスを使用するときマウスの種類を問合わせます．バスマウスのときは［1］［↵］，シリアルマウスのときは［2］［↵］です．

⑪

```
QuickBASIC　セットアップ　ユーティリティ                    Version 1.00

 ●数値演算ライブラリを選択してください[1:Alt Math / 2:Emulator Math]？1
 ●マウスを使用しますか[Y/N]？Y
 ●マウスの種類を選択してください[1:バスマウス / 2:シリアルマウス]1
 ●マウスドライバを選択して下さい[1:MOUSE.SYS  2:MOUSE.COM]？2

----------------------------------------------------------------------
 マウスドライバには、 CONFIG.SYS でデバイスドライバとしてメモリに組み込まれ
 る 1:MOUSE.SYS と 必要に応じて常駐、解放可能な 2:MOUSE.COM があります。
 どちらを利用してもかまいませんが、メモリ解放ができるため MOUSE.COM を選択
 されることをお勧めします。
 マルチプラン３．１をご使用の方は、1:MOUSE.SYS を必ず選択してください。
```

マウスドライバを入力します．通常は［2］［↵］です．

⑫

```
QuickBASIC　セットアップ　ユーティリティ                    Version 1.00
 ●数値演算ライブラリを選択してください[1:Alt Math / 2:Emulator Math]？1
 ●マウスを使用しますか[Y/N]？Y
 ●マウスの種類を選択してください[1:バスマウス / 2:シリアルマウス]1
 ●マウスドライバを選択して下さい[1:MOUSE.SYS  2:MOUSE.COM]？2
 ●ドキュメントファイルを転送しますか[Y/N]？Y

-----------------------------------------------------------------------
 QuickBASICには、README.DOC, SAMPLE.DOC, QBE.DOC など QuickB
 ASICをお使いになるうえで、重要なドキュメントファイルが含まれています。
 Yを選択した場合は、QBワークディスクもしくはハードディスクの DOC ディレク
 トリに転送されます。
 必ずお読みいただくようお願いします。
```

ドキュメントファイルのコピーをするかどうかを問合わせます．ドキュメントファイルはQuick BASICを使う上での注意事項等がかいてあります．コピーをする必要はありませんが，一読する必要はあります．既定値は Y ↵ です．

⑬

```
QuickBASIC　セットアップ　ユーティリティ                    Version 1.00
 ●数値演算ライブラリを選択してください[1:Alt Math / 2:Emulator Math]？1
 ●マウスを使用しますか[Y/N]？Y
 ●マウスの種類を選択してください[1:バスマウス / 2:シリアルマウス]1
 ●マウスドライバを選択して下さい[1:MOUSE.SYS  2:MOUSE.COM]？2
 ●ドキュメントファイルを転送しますか[Y/N]？Y
 ●サンプルプログラムを転送しますか[Y/N]？Y

-----------------------------------------------------------------------
 QuickBASICのパッケージには、多数のサンプルプログラムや便利なライ
 ブラリが含まれています。プログラミングの参考例やツールとしてご利用頂ければ
 幸いです。なお、サンプルプログラムの説明は、ドキュメントファイル SAMPLE.DOC
 に記述されています。
```

サンプルプログラムを転送するかどうかを問合わせます．サンプルプログラムの転送は必ずしも必要がありませんが，転送するとコマンド類の説明がわかりやすくなったり，サンプルプログラムを利

用したりすることができます．既定値は Y ↵ です．

⑭

```
QuickBASIC　セットアップ　ユーティリティ                     Version 1.00

 ●数値演算ライブラリを選択してください[1:Alt Math / 2:Emulator Math]？1
 ●マウスを使用しますか[Y/N]？Y
 ●マウスの種類を選択してください[1:バスマウス / 2:シリアルマウス]1
 ●マウスドライバを選択して下さい[1:MOUSE.SYS  2:MOUSE.COM]？2
 ●ドキュメントファイルを転送しますか[Y/N]？Y
 ●サンプルプログラムを転送しますか[Y/N]？Y


 ●上記の設定でよろしいですか[Y/N]？Y
-------------------------------------------------------------------------
 上記の設定でよければYを、変更したければNを指定してください。
 Nが指定された場合は、この画面の最初から指定しなおすことができます。
 この時、前回の設定がディフォルト値として表示されます。
```

以上の設定の確認を行います．良い場合は Y ↵ です．悪い場合は N ↵ です．N の場合は最初から入力しなおします．

⑮　続いて，次の表示に従って，”プログラムディスク”をＡドライブに，”起動ディスク”をＢドライブに入れ ↵ します．¥BIN¥QB.EXE 以下のファイルをＢドライブへコピーします．

```
QuickBASIC　セットアップ　ユーティリティ                     Version 1.00

『マスターディスク』
●ドライブ『A』に『 プログラムディスク 』をセットして下さい。

『ユーザーディスク』
●ドライブ『B』に『 QB起動ディスク 』をセットして下さい。

【何かキーを押して下さい】
```

⑯　以降は画面の指示でディスクを入れかえます．セットアップの手続きがおわると ④ へもどります

ので，3 ↵ でセットアップは終了です．

起動ディスクがQuick BASICを使うときのメインのディスクです．

⑰ バージョン4.2でのセットアップは次の手順となります．Quick BASICのプログラムディスクをBドライブに入れます．B:setup ↵ とします．まずQuick BASICセットアップユーティリティの説明が表示されます．続いてセットアップに関する質問が続きます．

セットアップを開始しますか[Y/N]? Y ↵

ハードディスクにセットアップしますか[Y/N]? N ↵

Quick BASICをどのドライブにインストールしますか? A ↵

マウスを使用しますか[Y/N]? Y ↵

マウスの種類を選択して下さい[1:バス/2:シリアル]? 1 ↵

マウスドライバを選択して下さい[1:MOUSE.SYS/2:MOUSE.COM]? 2 ↵

上記の設定でよろしいですか[Y/N]? Y ↵

以上によってセットアップが始まります．今回の例は上のようにしていますが自分の組合わせ設定を行います．この中の"Quick BASICをどのドライブにインストールしますか"はふだん使うワークディスクの入っているドライブを指定します．今回はワークディスクをAドライブに入れます．

後は画面の指示どおり行って下さい．

セットアップが終ります．

ワークディスクのみAドライブにのこします．今後はこれがQuick BASICを使うときのメインのディスクとなります．

⑱ なお，データを収納するディスクは，バージョン4.5でも4.2でもフォーマット（システムなし）ずみのディスクです．

【例題7】 プリンタのインストール

プリンタのインストールをせよ．

【解　説】 ◎ プリンタのインストールをします．

① DOSのバージョンによっては印刷ができないものがあります．次の表のとおりです．たとえば，DOSバージョン3.3ではこのままでは印刷ができません．次に印刷のできるようにします．

MS-DOS のバージョン		PRINT.SYS
MS-DOS Ver.2.0	(PS98-121-XXX)	不要
	(PS98-122-XXX)	〃
	(PS98-123-XXX)	〃
MS-DOS Ver.3.1	(PS98-125-XXX)	〃
	(PS98-127-XXX)	〃
	(PS98-129-XXX)	〃
	(PS98-011-XXX)	必要
MS-DOS Ver.3.3	(PS98-013-XXX)	〃

② MS-DOSのPRINT.SYSとPRINT.EXEのファイルをコピーします．DOSをAドライブに入れます．まずDIR A: ↵ によってPRINT.SYSおよびPRINT.EXEのファイルがあることを確認

します．別々のディスクに入っているものがあります．その場合はDOSのディスクをそのつど交換してください．まず，PRINT.EXEがあったとします．Bドライブにはバージョン4.5では起動ディスク4.2ではワークディスクを入れます．

COPY A:PRINT.EXE B: ⏎ です．

次にAドライブにPRINT.SYSの入っているディスクを入れます．

COPY A:PRINT.SYS B: ⏎ です．

これによって2個のファイルが起動ディスクまたはワークディスクへコピーされました．

③ 次にCONFIG.SYSの内容を変更します．

Aプロンプトから次の手順でCONFIG.SYSの内容をかきかえて保存しなおします．

A>QB ⏎

クイックベーシックの初期画面となります．

GRPH , F , O , C O N F I G . S Y S ⏎

次のCONFIG.SYSの内容が表示されます．

〔Ver.4.5〕

```
SHELL=A:¥COMMAND.COM A:¥ /P
FILES=20
BUFFERS = 10
```

〔Ver.4.2〕

```
FILES=20
BUFFERS = 20
```

カーソルを最後の行へおき次のようにかき加えます．

〔Ver.4.5〕

```
SHELL=A:¥COMMAND.COM A:¥ /P
FILES=20
BUFFERS = 10
DEVICE = PRINT.SYS
```

〔Ver.4.2〕

```
FILES=20
BUFFERS = 20
device = print.sys
```

GRPH , F , S によって，新しいCONFIG.SYSの内容が保存されました．これでプリンタが使えます．一度電源をおとします．

【例題8】 日本語フロントエンドプロセッサの組み込み

日本語フロントエンドプロセッサを組み込め．

【解 説】 ◎ 日本語フロントエンドプロセッサを組み込みます．

① 日本語を使うためには日本語フロントエンドプロセッサを組み込む必要があります．この手順は【例題7】で印刷のために関連のファイルを組み込み，CONFIGの内容をかきかえたのと同様です．

② ここではATOKを組み込みます．ATOKの入っている一太郎等のディスクを準備してください．DOSをAドライブに入れ電源を入れて，日，時を入力してDOSをたちあげAプロンプトとします．

AドライブにATOK6A.SYS，ATOK6B.SYSの2つのファイルのあるディスクを入れます．Bドライブにワークディスク（バージョン4.5でも4.2でもワークディスク）を入れます．

COPY A:ATOK6A.SYS B: ⏎

COPY A:ATOK6B.SYS B: ⏎

この2つによって2つのファイルがBドライブのディスクにかきこまれます.

③ なお，次のように1度でコピーすることもできます.

```
A>copy a:atok6*.* b:
A:ATOK6A.SYS
A:ATOK6B.SYS
        2 個のファイルをコピーしました.
```

④ 次にATOK.DICですが，これは漢字の辞書で非常に量が多いので起動またはワークディスクに入りません．したがって，データディスクにコピーします.

Aドライブに ATOK.DIC の入っているディスク，Bドライブにデータディスクを入れます.

COPY ATOK.DIC B: ↵ によりコピーされます.

```
A>copy a:atok.dic b:
        1 個のファイルをコピーしました.
```

⑤ 次にCONFIG.SYSをかきなおします．まずバージョン4.5ではATOKをワークディスクにかきましたので次のようにします．起動ディスク（バージョン4.5）をAドライブに入れ，電源を一度きって入れなおしてDOSをたちあげます．Aプロンプトから次のようにします.

A>Q B ↵

Quick BASICがたちあがります．Quick BASICの初期画面となります.

GRPH，F，O，C O N F I G . S Y S ↵ によりCONFIG.SYS の内容を表示させます．次のようにかき加えます.

```
DEVICE = B:ATOK6A.SYS / D = B / S = 1 / E = 1 / B = 0
DEVICE = B: ATOK6B.SYS
```

GRPH，F，S でこの内容が保存されます．これで起動ディスク（バージョン4.5）は完成です.

⑥ バージョン4.2ではワークディスクにATOKをかきましたので，次のようにします．ワークディスク（バージョン4.2）をAドライブに入れ，項⑤と同様にしてQuick BACICを立ち上がらせ，CONFIG.SYSを次のようにかきかえ保存します.

```
device = atok6a.sys / d = b / s = 1 / e = 1 / b = 0
device = atok6b.sys
```

⑦ なおこのディスクをそのままコピーしておくと，今後ディスクのセットアップをしなくてすみます.

【例題9】 Quick BASICのエディタモード

Quick BASICの初期画面エディタモードをつくれ.

【解　説】 ◎ Quick BASIC初期画面エディタモードをつくります.

① 起動ディスク（バージョン4.5）またはワークディスク（バージョン4.2）をAドライブに入れ電源を入れます．バージョン4.5を入れた場合はBドライブにワークディスク（バージョン4.5）を入れます．プリンタの使用，日本語ATOK6の使用，マウスの使用が可能となります．最後はAプロンプトです.

② Q B ⏎ とします．バージョン 4.5 では次の表示となり ESC をおすと初期画面となります．

```
F/ﾌｧｲﾙ  E/編集  V/表示  S/検索  R/実行  D/ﾃﾞﾊﾞｯｸﾞ  O/ｵﾌﾟｼｮﾝ          H/ﾍﾙﾌﾟ
                               Untitled

                              Welcome to

                 Microsoft (R) QuickBASIC Version 4.50J

             (C) Copyright Microsoft Corporation, 1985-1989.
                         All rights reserved.

                    現在，Easyメニューで起動しています

        <      RETURNキーで QB アドバイザが表示されます        >

              <      ESCキーで編集画面に移ります      >

                           ﾀﾞｲﾚｸﾄ ﾓｰﾄﾞ

<SHIFT+F1=ﾍﾙﾌﾟ> <F6=窓切換> <F2=SUB一覧> <F5=実行> <F8=ﾄﾚｰｽ> ¦        00001:001
```

バージョン 4.2 ではすぐ初期画面となります．これはプログラムの作成を行うエディットモードです．

〔Ver.4.2〕

```
F/ﾌｧｲﾙ  E/編集  V/表示  S/検索  R/実行  D/ﾃﾞﾊﾞｯｸﾞ  C/関数    H/ﾍﾙﾌﾟ(f･1)
                              <Untitled>

                             ﾀﾞｲﾚｸﾄ ﾓｰﾄﾞ

ﾒｲﾝﾌﾟﾛｸﾞﾗﾑ: <Untitled>    動作状況: 実行されていません           00001:001
```

③ CTRL ＋ XFER キーをおすと次のようになります．日本語が使えることがわかります．日本語を使うときは ATOK.DIC の入ったディスクを B ドライブに入れます．

F/ﾌｧｲﾙ　E/編集　V/表示　S/検索　R/実行　D/ﾃﾞﾊﾞｯｸﾞ　C/関数　H/ﾍﾙﾌﾟ（f･1）
<Untitled>

ﾀﾞｲﾚｸﾄ ﾓｰﾄﾞ

連ローマ字漢字

2章　プログラムの作成・ランタイム実行・保存・終了

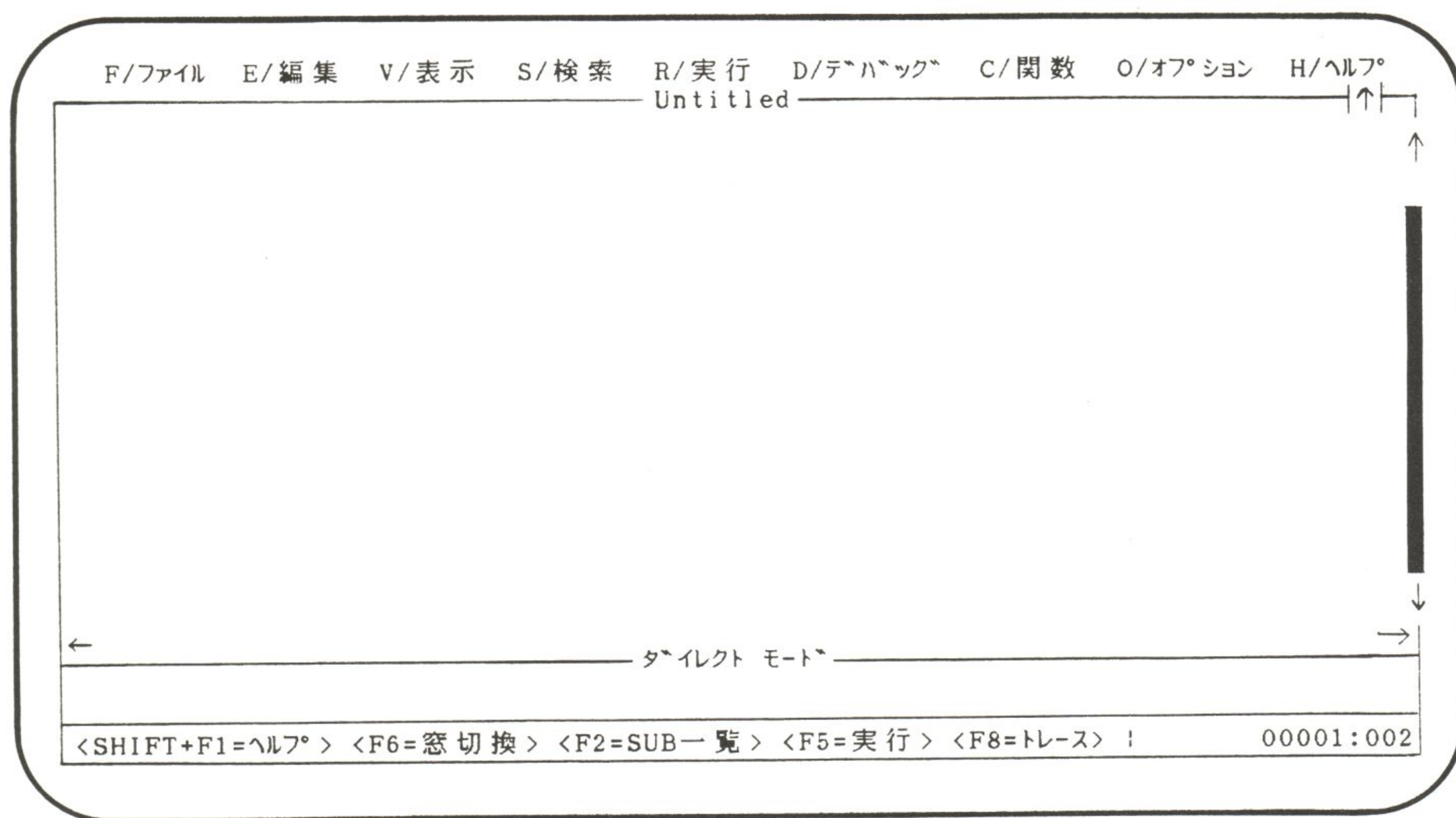
【例題 10】　テキスト画面にプログラムをかく

数 x，y を入力して，積を z に代入し，x の値*y の値＝z を表示するプログラムをつくれ.

【解　説】　◎　テキスト画面にプログラムをかきます.

① Quick BASIC をはしらせた画面は編集画面で，テキスト画面です．ここにプログラムを記入します．小文字で入力をします．何かすでに記入されているような場合はファイルメニューの"N/新規"を選びます.

〔Ver.4.5〕

```
F/ﾌｧｲﾙ  E/編集  V/表示  S/検索  R/実行  D/ﾃﾞﾊﾞｯｸﾞ  C/関数  O/ｵﾌﾟｼｮﾝ  H/ﾍﾙﾌﾟ
────────────────────── Untitled ──────────────────────

←                                                    →
──────────────────── ﾀﾞｲﾚｸﾄ ﾓｰﾄﾞ ────────────────────

<SHIFT+F1=ﾍﾙﾌﾟ> <F6=窓切換> <F2=SUB一覧> <F5=実行> <F8=ﾄﾚｰｽ> ¦   00001:002
```

② まず，x と y の入力ですから次のように入力します．行番号は必要ありません．これが従来の BASIC と大きく異なります.

```
input x,y
```

③ ⏎ とします．input が大文字となり，変数 x，y は小文字のままです．次のようになります.

```
INPUT x, y
```

④ 次に積を z に求めます．z＝x ＊ y です．次のようにかきます.

```
INPUT x, y
z=x*y
```

⑤ ⏎ をします．変数は小文字ですからそのままですが，見やすくスペースがとられます．

```
INPUT x, y
z = x * y
```

⑥ 次に表示の命令をかきます．次のようになります．

```
INPUT x, y
z = x * y
print x;"*";y;"=";z
```

⑦ ⏎ をおします．次のように print 命令は大文字となります．

```
INPUT x, y
z = x * y
PRINT x; "*"; y; "="; z
```

⑧ これで終了ですから end をかきます．

```
INPUT x, y
z = x * y
PRINT x; "*"; y; "="; z
end
```

⑨ ⏎ をおします．次のようになります．これでプログラムができました．

```
INPUT x, y
z = x * y
PRINT x; "*"; y; "="; z
END
```

【例題 11】 プログラムの保存

【例題 10】でつくったプログラムをプログラム名"renshu-1"でディスクに保存せよ．

【解　説】 ◎ ファイル（プログラム）を保存します．

① 【例題 10】でつくったプログラムをディスクに保存します．プログラムに誤りがあるかも知れないときでも，プログラムを実行して，重大な誤りのためコンピュータが暴走し，復帰できなくなることがあります．このときリセットするとメモリのプログラムは消去されますから，一応保存しておくことをおすすめします．完成したとき，再び同じ名前で保存しなおします．

B ドライブにはデータディスクを入れておきます．

② ファイルメニューを選択します．プログラムは Quick BASIC ではファイルとして扱いますので，今後はプログラム名ではなくファイル名と呼びます．ファイルはプログラムだけでなくデータの入っているファイルもありますから，ここではプログラムの入っているファイルと理解します．

③ "A/名前を変えて保存..." を選びます．次の画面となります．すなわち，ファイル名の入力，ファイルの保存形式の入力です．

〔Ver.4.5〕

```
F/ﾌｧｲﾙ  E/編集  V/表示  S/検索  R/実行  D/ﾃﾞﾊﾞｯｸﾞ  C/関数  O/ｵﾌﾟｼｮﾝ  H/ﾍﾙﾌﾟ
                        Untitled
                   A/名前を変えて保存
            ファイル名を設定してください

            N/ファイル名: ——— ファイル名をかく．Bドライブのときは B:ファイル名とする．

            A:¥

             D/ディレクトリ                                 TAB キーで項目間をカーソル
                                         ファイル形式      が動く．同じ項目の中は矢印キ
              BIN                   (*) Q/QuickBASIC       ーで選択
              LIB                         標準ファイル
              [-A-]
              [-B-]                 ( ) T/テキストファイル

                      < 確認 >  < 取消 >  < H/ヘルプ >

F1=ﾍﾙﾌﾟ  RETURN=実行  ESC=取消  TAB=次項目  ｶｰｿﾙｷｰ=選択  ¦    00001:001
```

④ まずファイル名を入力します．カーソルはN/ファイル名：の後にあります．

今回は renshu-1 ですから b : r e n s h u − 1 とします． ↵ をおします．b:は Bドライブのディスクに格納することを示します．省略するとカレントドライブへ入ります．

〔Ver.4.5〕

```
F/ﾌｧｲﾙ  E/編集  V/表示  S/検索  R/実行  D/ﾃﾞﾊﾞｯｸﾞ  C/関数  O/ｵﾌﾟｼｮﾝ  H/ﾍﾙﾌﾟ
                        Untitled
                   A/名前を変えて保存
            ファイル名を設定してください

            N/ファイル名:  b:renshu-1

            A:¥

             D/ディレクトリ
                                         ファイル形式
              BIN                   (*) Q/QuickBASIC
              LIB                         標準ファイル
              [-A-]
              [-B-]                 ( ) T/テキストファイル

                      < 確認 >  < 取消 >  < H/ヘルプ >

F1=ﾍﾙﾌﾟ  RETURN=実行  ESC=取消  TAB=次項目  ｶｰｿﾙｷｰ=選択  ¦    00001:001
```

⑤ 次に TAB キーをおして(*)Q/標準ファイル"*"にカーソルを移します． ↓ キーをおすと下の( )T/テキストファイルの( )の中に＊がつきます．再度 ↓ キーをおすともとへもどります．どちらか決めて ↵ します．

〔**Ver.4.5**〕

```
  F/ﾌｧｲﾙ  E/編集  V/表示  S/検索  R/実行  D/ﾃﾞﾊﾞｯｸﾞ  C/関数  O/ｵﾌﾟｼｮﾝ  H/ﾍﾙﾌﾟ
                                    RENSH-1.BAS
INPUT x, y
z = x * y
PRINT x; "*"; y; "="; z
END
```

Q/標準ファイルは名前のとおり標準ファイルです．T/テキストファイルはテキストファイルです．今回はどちらで格納してもかまいません．詳しくはMS-DOSの説明書をみてください．

renshu-1のファイルがBドライブのディスクにできます．このとき，拡張子 .BASがつきます．拡張子BASはBASICのプログラムを意味します．ファイル名が画面中央上部に表示されます．大文字で表示されますが，大文字，小文字の区別はありません．

【例題 12】　プログラムのランタイム実行

【例題 10】のプログラムをランタイムで実行せよ．

```
INPUT x, y
z = x * y
PRINT x; "*"; y; "="; z
END
```

【解　説】　◎　プログラムをランタイムで実行．

① プログラムの実行はランタイムの実行とEXEファイルを作成してDOSベースで実行する2つの方法があります．プログラムができたらランタイムで実行して，解答をみてプログラムのでき具合をチェックします．この結果によって，EXEファイルをつくるというのが普通の手順です．

② ランタイムの実行は，プログラムを簡単に実行する方法です．

③ プログラムが画面に表示されているものとします．

f･5 キーをおします．プログラムが実行されます．この場合は，数xとyの入力ですから?が表示されます．値を2つ入力します．値の間はカンマで区切ります．たとえば3，8です．↵ キーをおすと結果を表示します．

④ ”何かキーを押してください”と表示され，何かキーをおすと，元の画面にもどります．

```
ランタイムの実行
f・5 キー

? 3,8
 3 * 8 = 24

何かキーを押してください
```

⑤ 同じことを実行メニューからS/スタートを選ぶと，項③と同じことになります．

【例題 13】　EXE ファイルの作成

【例題 10】のプログラムを使って EXE ファイルをつくれ.

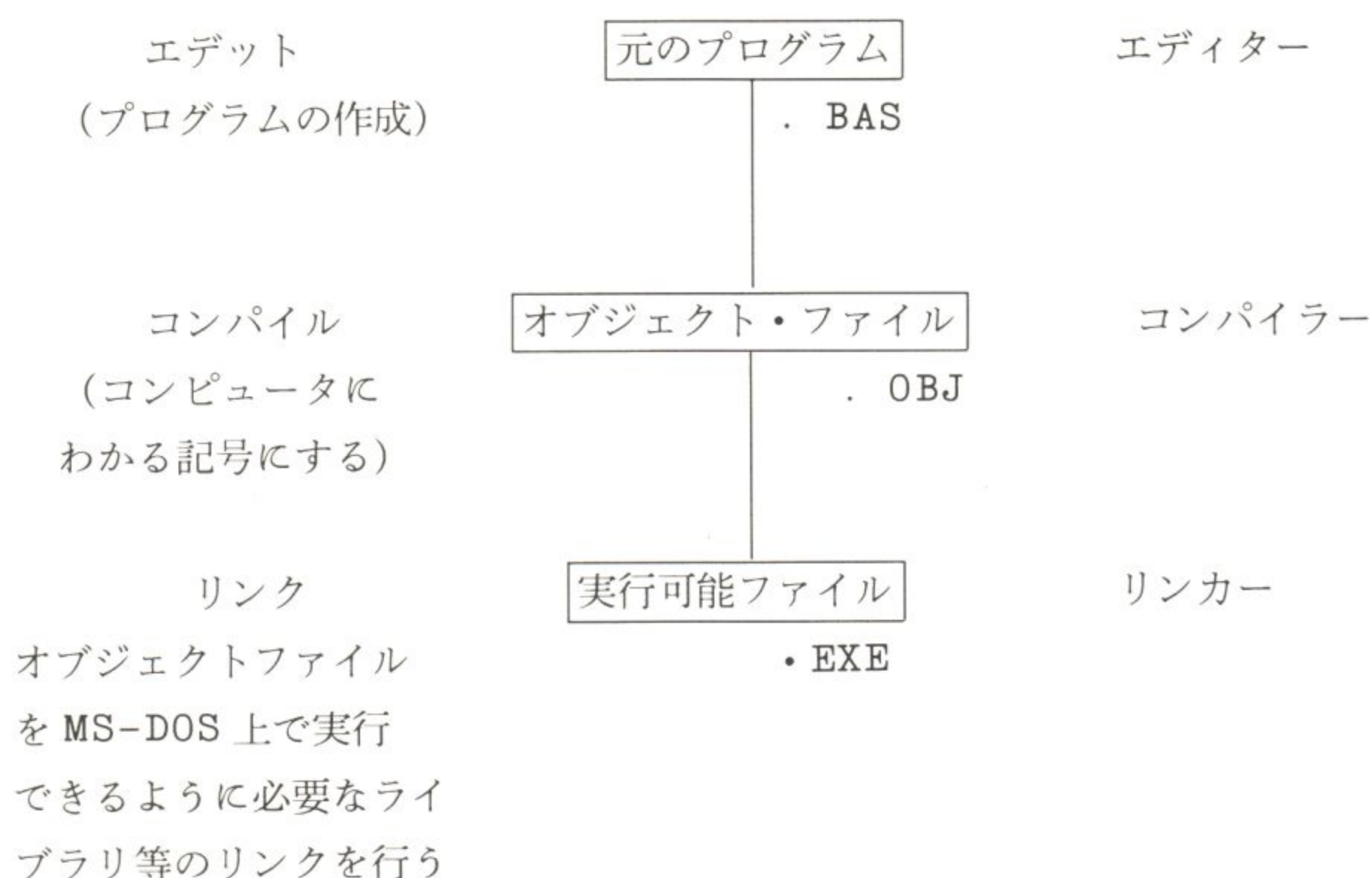
【解　説】　◎　EXE ファイルをつくります.

① 【例題 10】でつくったプログラムは【例題 11】で renshu-1.BAS として保存されます. このファイルをソースファイルと呼びます. BASIC でつくられたファイルですから拡張子に BAS がつきます.

② 次にこれをコンパイルしてオブジェクトファイルをつくります. この作業を行う機能を持つソフトウェアをコンパイラーと呼びます. できたファイルは OBJ の拡張子がつきます.

③ 次にオブジェクトファイルをリンクして実行可能なファイルをつくります. これを実行可能ファイルあるいは EXE ファイルと呼びます. この作業を行う機能を持つソフトをリンカーと呼びます. EXE ファイルは拡張子として EXE がつきます.

④ この EXE ファイルが MS-DOS 上で実行できるファイルとなります. これらのすべての機能が Quick BASIC には備わっています. この環境を称して統合環境といいます. プログラム作成の一連の流れは次のとおりです.

エデット (プログラムの作成)	元のプログラム .BAS	エディター
コンパイル (コンピュータにわかる記号にする)	オブジェクト・ファイル .OBJ	コンパイラー
リンク オブジェクトファイルを MS-DOS 上で実行できるように必要なライブラリ等のリンクを行う	実行可能ファイル ・EXE	リンカー

⑤ では手順を追っていきましょう. まず, プログラムが画面に表示されているものとします.

⑥ 実行メニューを選択します.

⑦ X/EXE ファイル作成... を選択します. 次の画面となります. この画面ではコンパイルスイッチ (バージョン 4.2 では作成形式) を決めます. 今回は X/ランタイム分離型の前の( )内に＊マークがありますので ⏎ します. ランタイム分離型 EXE ファイルをつくります.

名前を変えたいときは N/実行ファイル名: (バージョン 4.2 では N/EXE ファイル名:) にカーソルがあるときその名前を入れます. たとえば RR.EXE あるいは RR です. 拡張子 EXE は自動的につけられます. ⏎ をすると実行を始めます.

TAB キーを使うとカーソルが項目間を移動しますので指定をします. たとえばコンパイルス

イッチ（バージョン 4.2 では作成形式）では(＊)X/ランタイム分離型にカーソルが移りますので，↓キーで( )A/独立型を指定することができます．M/実行ファイル作成（バージョン 4.2 では M/EXE ファイル作成）を選んで ↵ すれば作成に入ります．E/実行ファイル作成後終了（バージョン 4.2 では E/EXE ファイル作成後終了）では EXE ファイルをつくってから MS-DOS へもどります．取消で ↵ では元の画面にもどります．

〔Ver.4.5〕

```
 F/ファイル  E/編集  V/表示  S/検索  R/実行  D/デバッグ  C/関数  O/オプション  H/ヘルプ
                              RENSHU-1.BAS
INPUT x, y
z = x * y
PRINT x; "*"; y; "="; z
END                           X/実行ファイル作成
              実行ファイル名とコンパイルスイッチを設定してください

              N/実行ファイル名:  b:renshu-1.exe

              [ ] D/デバッグ                   (*) X/ランタイム分離型
                                               ( ) A/独立型

              < M/実行ファイル作成 >  < E/実行ファイル作成後終了 >
                                         < 取消 >  < H/ヘルプ >
```

⑧ コンパイルとリンクをして，EXE ファイルができます．終了すると元の画面へもどります．

【例題 14】　ファイルの確認

【例題 13】でつくったファイルを MS-DOS 上で確認せよ．

【解　説】　◎　.EXE ファイルを確認します．

① 【例題 13】でつくった EXE ファイルを確認します．EXE ファイルは MS-DOS 上で動くファイルです．

② Quick BASIC が使われている状態の場合は MS-DOS へもどして確認します．Quick BASIC では MS-DOS へもどす方法は 2 つあります．第 1 の方法はファイルメニューから X/終了を選びます．MS-DOS の A>となります．Quick BASIC にもどすには"QB"によります．第 2 の方法は一時的に MS-DOS へもどす方法で，ファイルメニューから D/MS-DOS コマンドを選び一時的に MS-DOS のコマンドを使えるようにします．Quick BASIC には Exit ↵ と何かキーをおすともどります．

③ ここではファイルメニューから D/MS-DOS コマンドで MS-DOS コマンドを使いましょう．

④ MS-DOS のコマンドを利用して次のようにします．

DIR ⏎ によってカレントドライブのファイルのリストができます．DIR/W ⏎ 内で画面いっぱいにリストがでます．

DIR B: ⏎ でBドライブのファイルのリストがでます．DIR B:/W⏎ でBドライブのリストが画面い:ぱいにでます．B:のかわりにA:とすればAドライブの内容です．

カレントドライブとはMS-DOSが現在使用しているドライブでA>のときはAドライブ，B>のときはBドライブです．A>でB: ⏎ とすればAドライブからBドライブへカレントドライブを変更できます．

⑤ Renshu-1.EXEとRenshu-1.OBJ，Renshu-1.BASがあることを確認して下さい．

> **【例題15】 MS-DOSでEXEファイルの実行**
>
> 【例題13】でつくったファイルRenshu-1.EXEを実行せよ．

【解　説】 ◎ EXEファイルを実行します．

① Quick BASIC画面のファイルメニューからM/MS-DOSコマンドを選んでMS-DOSコマンドへ入ります．

renshu-1 ⏎ によりrenshu-1.EXEファイルが実行されます．

② 数値を2つ入力すると値が求められます．

③ EXIT ⏎，および何かキーをおすとQuick BASICへもどります．

④ MS-DOS上ではrenshu-1 ⏎ で同様の実行ができます．

MS-DOSコマンドモートでの実行
実行後Quick BASICへもどる画面

```
A>b:renshu-1                          実行
? 56,34                               56と34を入力
 56 * 34 = 1904

EXIT で QuickBASIC に戻ります


Command ﾊﾞｰｼﾞｮﾝ 3.30

 A>exit                               Quick BASICへもどる
```

MS-DOSでの実行

```
A>b:renshu-1                 実行
? 56,34                      56と34を入力
 56 * 34 = 1904

A>                           Quick BASICへもどるにはQB⏎とする.
```

> **【例題16】 終　了**
>
> 【例題10】のプログラムをつくり実行し，DOSへもどれ．

【解　説】　◎　DOSへもどります.

① ファイルメニューから"X/終了"を選びます.

Quick BASICを終了しDOSにもどります. Quick BASICへもどるにはQB ⏎ から始めます.

3章　プログラムの読み込み・修正・保存・サブルーチンとファンクションの作成・印刷

【例題 17】　ファイルの呼び出し

ファイル renshu-1.BAS をディスクから呼び出せ.

【解　説】　◎　ファイルをディスクから呼び出します.

① ファイルメニューを選びます.

② O/読込を選びます. 次の画面となります.

〔Ver.4.5〕

```
F/ﾌｧｲﾙ  E/編集  V/表示  S/検索  R/実行  D/ﾃﾞﾊﾞｯｸﾞ  C/関数  O/ｵﾌﾟｼｮﾝ  H/ﾍﾙﾌﾟ
                              Untitled
                                O/読込
    ファイル名を設定してください

    N/ファイル名:  *.BAS

    A:¥
              F/ファイル                        D/ディレクトリ

                                                 BIN
                                                 LIB
                                                 [-A-]
                                                 [-B-]



                        < 確認 >  < 取消 >  < H/ヘルプ >
```

ファイル名をかく. B ドライブにしまうときは B:ファイル名とする.

③ ファイル名に B:Renshu-1.BAS とタイプして ↵ します. B ドライブより Renshu-1.BAS を呼び出します. ドライブを指定するときは a:Renshu-1, b:Renshu-1 です. BAS は省略できます. 大文字, 小文字の区分はありません.

N/ﾌｧｲﾙ名:　　renshu-1

④ ファイルがすでにある場合は, TAB キーをおすとファイルリストの項目へカーソルが移りますので, 矢印キーを使って希望のファイル名をさがしてカーソルをおき ↵ しても同じです.

⑤ ファイルがないと「ファイルが見つかりません」の表示がされます. ↵ キーで元へもどります.

【例題 18】 プログラムの修正

【例題 10】のプログラムを修正して x, y, z を入力して，積を求めて結果を表示するプログラムをつくれ．ただし，積は変数 zz に入れる．

【解　説】 ◎ プログラムの修正をします．

① 【例題 17】の方法でファイル Renshu-1.BAS を呼び出します．これを元にして，プログラムの変更をします．

```
INPUT x, y
z = x * y
PRINT x; "*"; y; "="; z
END
```

② 1行目の y の後に，z を追加します．

カーソルを y の後におきます．,と z をタイプします．[↵] はおしません．z をタイプした後[↵]をおすと次に1行空白行ができます．[DEL] キーをおせば元にもどります．

③ 次に z=x*y の z を zz とします．

カーソルを z または z の後におき z とタイプします．z=x*y が zz=x*y となります．Quick BASIC の初期状態では挿入モードとなっていますので z にカーソルをおき z をおせば z が挿入されます．たとえば abc の b にカーソルをおき xyz とタイプすると axyzbc となります．

挿入モードでないとき z とすると z が消えて z がかかれます．abc で b にカーソルをおき xy とすると axy となります．挿入モードはカーソルをよくみると_（アンダースコア）の点滅となっています．挿入モードでないときはカーソルが四角の反転で点滅となっています．挿入モードは [INS] キーをおすことで切りかえられます．

④ zz=x*y の後に*z を追加します．y の後にカーソルをおき＊ z とします．[↵] はしません．

⑤ PRINT 文を修正します．次の①，②，③の手順とします．

```
PRINT x;"*";y;"=";z
```

③カーソルを [↓] または [↑] キーで移す．

②カーソルを移し z とする．

①カーソル，挿入モード
"*";z;とする．[↵] はおしません．[↵] をおすと"以降が次行へうつります．
[BS] キーをおすと元へもどります．

⑥ 文字の消去は [DEL] キーでカーソルのある文字，[BS] キーでカーソルの前の文字が消えます．

⑦ これでプログラムができました．ファイルの保存は【例題 19】で行います．

```
INPUT x, y, z
zz = x * y * z
PRINT x; "*"; y; "*"; z; "="; zz
END
```

【例題 19】　同名でファイルを保存

【例題 18】で修正してつくったプログラムを RENSHU-1.BAS のまま保存せよ.

【解　説】　◎　同名でファイルを保存します.

① ファイルメニューから"S/保存"を選びます.

修正されたプログラムが現在のファイル名 RENSHU-1.BAS で保存されます. 確認の問合わせはありませんから注意してください. 修正前のプログラムはなくなります.

【例題 20】　サブルーチンの作成（1）

次のプログラムをつくれ. このプログラムは x と y を入力して x^2+y^2 を求め, その値をサブルーチン a へ引き渡して表示するものです.

```
DECLARE SUB a (x)
INPUT x, y
z = x * x + y * y
CALL a(z)
END


SUB a (x)
PRINT x
END SUB
```

（実行例）

```
? 2,3            2, 3 を入力した例
 13
```

【解　説】　◎　サブルーチンをつくります.

① まずメインルーチンを次のようにかきます. DECLARE コマンドはファイルへしまうとき自動的にかかれますので, かく必要はありません.

```
INPUT x, y
z = x * x + y * y
CALL a(z)
END
```

② 次にサブルーチンをつくります.

編集メニューを選び"S/新規 SUB..."を選びます.

③ 続いて, SUB の名前を聞いてきます. [a] [↵] とします.

〔Ver.4.5〕

```
  F/ファイル  E/編集  V/表示  S/検索  R/実行  D/デバッグ  C/関数  O/オプション  H/ヘルプ
                                  Untitled
INPUT x, y
z = x * x + y * y
CALL a(z)
END

                                  S/新規 SUB
                    作成するプロシージャ名を入力してください

                    N/名前：  a

                         < 確認 > < 取消 > < H/ヘルプ >
```

④　次のように SUB を表示します.

```
SUB a
END SUB
```

⑤　SUB a の後に仮引数の(x)をかきます. 次行には, PRINT x をかきます. サブルーチンができました.

```
SUB a (x)
PRINT x
END SUB
```

⑥　なお, SHIFT + f·2 をおすことでメインルーチンとサブルーチンの間を交互にみることができます.

⑦　メインルーチンの x とサブルーチンの x は各々独立しています. これは後ほど詳しく説明します.

【例題 21】　サブルーチンの作成 (2)

次のサブルーチンを含むプログラムをつくれ.

```
DECLARE SUB a (x!)
INPUT x, y
z = x * y
CALL a(z)
END


SUB a (x)
PRINT x
END SUB
```

【解　説】　◎　サブルーチンを含むプログラムをつくります.

①　Quick BASIC ではサブルーチン SUB, ファンクション FUNCTION を使うことができます.

SUB を例にサブルーチンを含むプログラムのつくり方を示します．なおプログラムの使い方については I 編 20 章で詳しく扱います．

② 次のようにプログラムをかきます．

```
INPUT x, y
z = x * y
CALL a(z)
END
```

③ 次に最後の行に SUB a(x) とかきます．

```
INPUT x, y
z = x * y
CALL a(z)
END
sub a(x)
```

④ ↵ またはカーソルを別の行へ移すと次のようにサブルーチン画面となります．

```
SUB a (x)

END SUB
```

⑤ カーソルを中間のスペース行におき，PRINT 文をかきます．

```
SUB a (x)
PRINT x
END SUB
```

⑥ これでプログラムはできあがりです．先頭の"DECLARE SUB a(x!)"は自動的につけられます．

⑦ メインプログラムの画面とサブルーチンの画面の切りかえは SHIFT + f・2 キーで行います．

⑧ これを RRR-95 の名前で保存します．今回はテキストファイルで保存します．

【例題 22】　モジュールの追加

【例題 21】のプログラム（ファイル名 RRR-95）を修正して次のプログラムとせよ．ただし，RRR-95 のプログラムを追加してから修正していくものとする．

プログラム例

```
DECLARE SUB a1 (z1!)
DECLARE SUB a (x!)
INPUT x, y
z = x * y
CALL a(z)
INPUT x1, y1
z1 = x1 / y1
CALL a1(z1)
END


SUB a (x)
PRINT x
END SUB


SUB a1 (z1)
PRINT z1
END SUB
```

実行結果例

```
? 5,3        5 と 3，2 と 3 を入力の例
 15
? 2,3
 .6666667
```

【解　説】　◎ ファイルの追加をします．

① 【例題 21】のプログラムの"END"の部分にカーソルをおきます.

② ファイルメニューを選択します. 次に"M/結合..."(バージョン 4.5), M/追加..."(バージョン 4.2) を選びます.

③ 追加するファイル名を問合わせます. RRR-95 を指定します.

```
N/ﾌｧｲﾙ名:        b:rrr-95

B:¥
```

④ ⏎ により次のように RRR-95 の内容が追加されます. なお, ここではサブルーチンがすでに存在するものと同名になりますので図のような警告がでますが, 修正しますのでこれは無視して ⏎ をおします. ESC キーをおします. メインプログラムの画面となります.

〔Ver.4.2〕

```
F/ﾌｧｲﾙ  E/編集  V/表示  S/検索  R/実行  D/ﾃﾞﾊﾞｯｸﾞ  C/関数    H/ﾍﾙﾌﾟ(f・1)
                              RRR-95.BAS
DECLARE SUB a (x!)
INPUT x, y
z = x * y
CALL a(z)
DECLARE SUB a (x!)        追加ファイル
INPUT x, y
z = x * y
CALL a(z)
END                  同名のｻﾌﾞﾌﾟﾛｸﾞﾗﾑを読み込もうとしました      警告
                                  a                        同名のサブルーチン
SUB a (x)                                                  が存在する
PRINT x
END SUB                          確認

END
```

〔Ver.4.5〕

```
                 同名のプロシジャーを読み込もうとしました      警告
                                   a                         同名のプロシジュアが
                                                             存在する
                       < 確認 >    <H/ヘルプ>
```

⑤ バージョン 4.5 とバージョン 4.2 では若干結合・追加のされ方が違いますが, いずれも表示を利用して, 問題のプログラムのように修正します.

〔Ver.4.5〕

```
F/ﾌｧｲﾙ  E/編集  V/表示  S/検索  R/実行  D/ﾃﾞﾊﾞｯｸﾞ  C/関数  O/ｵﾌﾟｼｮﾝ  H/ﾍﾙﾌﾟ
                              RRR-95.BAS
DECLARE SUB a (x!)
INPUT x, y
z = x * y
CALL a(z)
END

DECLARE SUB a (x!)
INPUT x, y
z = x * y
CALL a(z)
END

SUB a (x)
PRINT x
END SUB
```

【例題23】 ファンクション

次のプログラムをつくれ. このプログラムはxを入力し, ファンクションfにxを引きわたし, その3乗を求めて表示するものとする.

```
DECLARE FUNCTION f! (z!)          ? 3        3を入力の例
INPUT x                            27
PRINT f(x)                        ? 10       10を入力の例
END                                1000


FUNCTION f (z)
f = z * z * z
END FUNCTION
```

【解 説】 ◎ ファンクションをつくります.

① 問題のメインルーチンをかきます. DECLAREコマンドは自動的につけられますからかく必要はありません. FUNCTIONについては21章で詳しく学びます.

② 編集メニューを選択します. ここで"F/新規FUNCTION ..."を選びます.

③ 次の画面となります. 名前を入力します.

```
N/名前: f
```

④ ⏎により次の画面となります.

```
FUNCTION f
END FUNCTION
```

⑤ FUNCTIONを次のようにかきます. これでできあがりです.

```
FUNCTION f (z)
f = z * z * z
END FUNCTION
```

【例題24】 ファイルの印刷

【例題21】のファイルを印刷するプログラムをつくれ.

【解 説】 ◎ ファイルの印刷をします.

① 【例題21】のプログラムが表示されているものとします.

② ファイルメニューからP/印刷 ...を選びます. 次の画面となります.

〔Ver.4.5〕

```
P/印刷
印刷する範囲を選択してください

( ) S/指定範囲
( ) W/現在のウィンドウ
(*) M/現在のモジュール
( ) A/全体

< 確認 > < 取消 > < H/ヘルプ >
```

〔Ver.4.2〕

```
印刷
( ) S/指定範囲
( ) W/アクティブウインドウ
(*) M/カレントモジュール
( ) A/全体

確認    取消
```

③　印刷の種類を 4 つの中で選びます．矢印キーを使います．既定値は M/現在のモジュールでアクティブウインドウのモジュールです．サブルーチンがあればサブルーチンも印刷します．S/指定範囲はカーソルのある行を印刷します．W/アクティブウインドウはアクティブウインドウ全体です．つまり，現在表示されているウインドウのみです．A/全体はメモリーのモジュール，インクルードファイル，ドキュメントファイルすべてを印刷します．

【結　果】

M/カレントモジュールの例

アクティブウインドウ

```
DECLARE SUB a (x!)
INPUT x, y
z = x * y
CALL a(z)
END
```

サブルーチン

```
SUB a (x)
PRINT x
END SUB
```

A/全体の例

アクティブウインドウ

```
DECLARE SUB a (x!)
INPUT x, y
z = x * y
CALL a(z)
END
```

サブルーチン

```
SUB a (x)
PRINT x
END SUB
```

W/アクティブウインドウの例

アクティブウインドウ（サブルーチンは印刷しない）

```
INPUT x, y
z = x * y
CALL a(z)
END
```

W/アクティブウインドウの例

アクティブウインドウ（メインプログラムは印刷しない）

SHIFT ＋ f･2 でメインプログラムとサブルーチンのウインドウを切りかえる．

```
SUB a (x)
PRINT x
END SUB
```

S/指定範囲の例

z にカーソルをおいた場合

```
z = x * y
```

4章　ダイレクト実行

【例題 25】　ダイレクトモード

x 度を入力して，12*SIN(x 度)を求めて表示するプログラムをつくり，実行前に SIN(15 度)の値をダイレクトモードで確認せよ．

プログラム例

```
INPUT x
PRINT 12 * SIN(x / 180 * 3.14159)
END
```

15 を入力の結果例

```
? 15
 3.105826
```

【解　説】　◎　ダイレクトモードで直接命令を実行します．

① プログラムをつくった後，[f･6] によりダイレクトモードをアクティブ画面にします．

② ダイレクト画面に次のように書き込みます．

```
INPUT x
PRINT 12 * SIN(x / 180 * 3.14159)
END
```

```
                                   ダイレクト モード
print sin (15/180*3.14159)
```

③ [↵] により 15 度の SIN の値を求めることができます．

```
.2588189
```

【例題 26】　ダイレクトモードで確認してからプログラムへ挿入

【例題 25】と同じプログラムをつくれ．ただし，2 行目の PRINT 文は，ダイレクトモードで 15 度の SIN を求める命令を書き実行して，確認してから，挿入して修正してプログラムを完成するものとする．

【解　説】 ◎ ダイレクトモードで命令を確認してからプログラムを挿入します.

① 次のように2行を記入したところで，[f･6] によりダイレクトモードをアクティブ画面とします. ここで15度のSINを求める式をかき実行し確認します.

```
INPUT x
END
```

ﾀﾞｲﾚｸﾄ ﾓｰﾄﾞ

```
print sin (15/180*3.14159)
```

② "print sin(15/180*3.14159)"にカーソルをおき，[CTRL] + [Y] をおします. この命令がクリップボードにコピーされます. 編集メニューでP/ペーストを選ぶと再びこの命令がカーソルのあるダイレクトモードの1行目にかかれます. [f･6] によりアクティブ画面をかえ"END"にカーソルを移し，編集メニューからP/ペーストを選んでこの命令を"END"の前に挿入します.

```
INPUT x
PRINT SIN(15 / 180 * 3.14159)
END
```

ﾀﾞｲﾚｸﾄ ﾓｰﾄﾞ

```
print sin (15/180*3.14159)
```

③ "12*"の挿入と15をxに変更してプログラムを完成します.

```
INPUT x
PRINT 12 * SIN(x / 180 * 3.14159)
END
```

ﾀﾞｲﾚｸﾄ ﾓｰﾄﾞ

```
print sin (15/180*3.14159)
```

【例題 27】　ダイレクトモードでサブルーチンの実行

次のプログラムをつくり，次にサブルーチン a1，a2，a3 を各々独立して実行せよ．このプログラムは x 度を入力して，x 度の sin，cos，tan をサブルーチンを使って求めるものです．

```
DECLARE SUB a1 (a1)
DECLARE SUB a2 (a2)
DECLARE SUB a3 (a3)
INPUT x
CALL a1(x)
CALL a2(x)
CALL a3(x)
END


SUB a1 (z1)
    PRINT SIN(z1 / 180 * 3.14159)
END SUB

SUB a2 (z2)
    PRINT COS(z2 / 180 * 3.14159)
END SUB

SUB a3 (z3)
    PRINT TAN(z3 / 180 * 3.14159)
END SUB
```

```
? 25              25を入力した結果例
 .422618
 .9063079
 .4663073
```

【解　説】　◎　サブルーチンを独立してダイレクトに実行します．

① プログラムをかきます．

② f･6 でアクティブ画面をダイレクトモードとします．

次のように call 命令をかきます． ↵ によりサブルーチンのみ実行します．

CALL a1(30) ↵ で 30 度の sin の値を求めます．

CALL a2(75) ↵ で 75 度の cos の値を求めます．

CALL a3(45) ↵ で 45 度の tan の値を求めます．

```
INPUT x
CALL a1(x)
CALL a2(x)
CALL a3(x)
END
```

```
                              ダイレクト モード

call a1(30)
call a2(75)
call a3(45)
```

ダイレクト実行の結果

```
.4999996
.2588201
.9999988
```

【例題 28】 変数の値を変えて実行

1から1000までを表示するプログラムをつくり，途中 STOP キーでプログラムを中断し，変数の値を変えて，そこから再開して1000まで表示せよ．

【解 説】 ◎ 変数の値をダイレクトモードで変更します．

① 次のプログラムをつくり， f·5 で実行します．次に STOP キーをおして途中で中断します．ここで f·6 キーでアクティブ画面をダイレクトモードとして，i=950 ↵ とします．中断したときの i の値に関係なく i は 950 となります．

```
FOR i = 1 TO 1000
    PRINT i;
NEXT i
PRINT
END
```

```
                              ﾀﾞｲﾚｸﾄ ﾓｰﾄﾞ
i=950
```

② f·5 で再開します．i が1000まで実行されてプログラムが終了します．

③ 次の例はプログラムを実行し， STOP キーをおしたとき i は 179 でそれを表示したあと NEXT i の実行前で止まりました．"NEXT i"が緑色になっています．i=950 をダイレクト実行して再開しましたので，NEXT i により i は 951 となり 951 から1000まで表示します．

【結 果】

```
 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22
23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42
43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62
63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82
83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  10
2  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117  11
8  119  120  121  122  123  124  125  126  127  128  129  130  131  132  133  13
4  135  136  137  138  139  140  141  142  143  144  145  146  147  148  149  15
0  151  152  153  154  155  156  157  158  159  160  161  162  163  164  165  16
6  167  168  169  170  171  172  173  174  175  176  177  178  179  951  952  95
3  954  955  956  957  958  959  960  961  962  963  964  965  966  967  968  96
9  970  971  972  973  974  975  976  977  978  979  980  981  982  983  984  98
5  986  987  988  989  990  991  992  993  994  995  996  997  998  999  1000
```

5章　サブファイル・テキストファイル・メインプログラムの設定・インクルードファイル

【例題 29】　サブファイル

次の 2 つのファイルをつくり，各々，RRR-71.BAS，RRR-72.BAS で保存する．次に，RRR-71 のサブルーチン plus と RRR-72 のサブルーチン multi を呼び出して，入力した値までの 1 からの和と階乗を求めるプログラム RRR-73.BAS をつくれ．なお，RRR-71 は値を入力して，1 からその値までの和を求めるプログラム，RRR-72 は値を入力して，階乗を求めるプログラムとする．

RRR-71.BAS のプログラムと 6 を入力の結果

```
DECLARE SUB plus (z%, zz%)
INPUT x%
CALL plus(x%, y%)
PRINT "1 ｶﾗ "; x%; "ﾏﾃﾞﾉ ﾜ="; y%
END

SUB plus (z1%, z2%)
FOR i = 1 TO z1%
    y% = y% + i
NEXT i
z2% = y%
END SUB

? 6
1 ｶﾗ  6 ﾏﾃﾞﾉ ﾜ= 21
```

RRR-72.BAS のプログラムと 6 を入力の結果

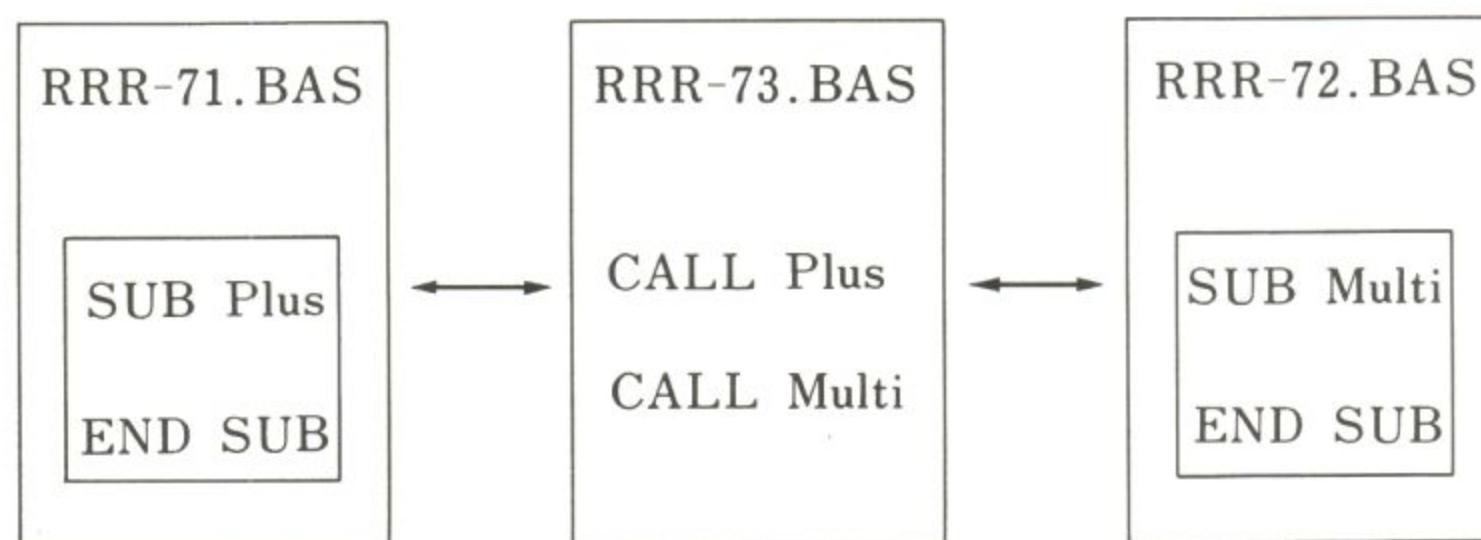
```
DECLARE SUB multi (z%, zz%)
INPUT x%
CALL multi(x%, yy%)
PRINT "1 ｶﾗ "; x%; "ﾏﾃﾞﾉ ｶｲｼﾞｮｳ="; yy%
END

SUB multi (z1%, z2%)
yy% = 1
FOR i = 1 TO z1%
    yy% = yy% * i
NEXT i
z2% = yy%
END SUB

? 6
1 ｶﾗ  6 ﾏﾃﾞﾉ ｶｲｼﾞｮｳ= 720
```

RRR-73.BAS のプログラム

```
INPUT x%
CALL plus(x%, y%)
CALL multi(x%, yy%)
PRINT y%
PRINT yy%
END
```

【解　説】 ◎ サブファイルを利用します.

① Quick BASICでは複数のファイルをロードして，別々に管理するとともに，別のファイルのモジュールをメインモジュールから利用することができます．今回の例は次のようなものです.

RRR-71.BAS		RRR-73.BAS		RRR-72.BAS
SUB Plus END SUB	←→	CALL Plus CALL Multi	←→	SUB Multi END SUB

② RRR-71.BAS, RRR-72.BAS, RRR-73.BASを各々作成し，保存します.

③ ファイルメニューから"L/サブファイル読込 ... (バージョン 4.5)", L/サブファイル常駐 ..." (バージョン 4.2) を選びます.

④ ファイル名を問合わせてきますのでrrr-71を入力します．なお形式は"M/モジュール"とします.

〔Ver.4.5〕

```
 F/ファイル  E/編 集   V/表 示   S/検 索   R/実 行   D/デバッグ   C/関 数   O/オプション  H/ヘルプ
                                 RRR-71.BAS
DECL                          L/サ ブ フ ァ イ ル 読 込
INPU   フ ァ イ ル 名 ， フ ァ イ ル 形 式 を 設 定 し て く だ さ い
CALL
PRIN   N/フ ァ イ ル 名：  rrr-71
END
       B:¥
                        F/フ ァ イ ル                         D/デ ィ レ ク ト リ

         RENSHU-1.BAS  RRR-95.BAS                              [-A-]
         RENSHU-2.BAS  RRR-96.BAS                              [-B-]
         RRR-100.BAS   RRR-97.BAS
         RRR-71.BAS    RRR-98.BAS
         RRR-72.BAS    RRR-99.BAS
         RRR-73.BAS

             (*) M/モ ジ ュ ー ル プ ロ グ ラ ム
             ( ) I/イ ン ク ル ー ド フ ァ イ ル
             ( ) D/ド キ ュ メ ン ト フ ァ イ ル

                           < 確 認 >  < 取 消 >  < H/ヘ ル プ >
```

⑤ 同様に，ファイルメニューから[L]，rrr-72[↵]，サブファイルから[L]，rrr-73[↵]により3つのファイルをメモリへ呼びます.

⑥ 実行メニューから"M/メインモジュールの設定 ..."を選びます.

⑦ メインプログラムの設定画面となります．ここでは実行するのはrrr-73ですから，カーソルをrrr-73へ移し，[↵]します．rrr-73がメインプログラムとなります.

〔Ver.4.5〕

```
        M/メインモジュールの設定
メインモジュールを選択してください


  RRR-73.BAS
  RRR-71.BAS
  RRR-72.BAS
```

〔Ver.4.2〕

```
メインプログラムの選択：

  RRR-71.BAS
  RRR-72.BAS
  RRR-73.BAS
```

⑧ [f･2] をおすと次のようにメインプログラムが rrr-73 と確認できます.

〔Ver.4.5〕

```
                          S/SUB一覧
C/編集するプロシージャ，ウィンドウを選択してください


  RRR-73.BAS
  RRR-71.BAS
    plus
  RRR-72.BAS
    multi




RRR-73.BAS:=メインモジュール
```

〔Ver.4.2〕

```
C/プログラム選択：

  RRR-73.BAS
  RRR-71.BAS
    plus
  RRR-72.BAS
    multi




RRR-73.BAS:=メインプログラム
```

⑨ [f･5] で実行します. rrr-73 のメインプログラムが, rrr-71 のサブルーチン plus と rrr-72 のサブルーチン multi を使って結果を求めます.

【結　果】

```
? 6        (6を入力の例)
 21
 720
```

> **【例題 30】　サブファイルの開放**
>
> 【例題 29】の状態から rrr-71, rrr-72 を開放せよ.

【解　説】　◎　サブファイルを開放します.

① 【例題 29】の手順で, サブファイル rrr-71, rrr-72 をメモリへ読み込みます.

② ファイルメニューを選びます.

③ "U/サブファイル開放 ..."を選びます. メモリに常駐しているファイル名が一覧されます. まず, rrr-72.BAS にカーソルを移し ⏎ します.

〔Ver.4.5〕

```
U/サブファイル解放
解放するモジュールを選択してください

RRR-73.BAS
RRR-71.BAS
RRR-72.BAS
```

〔Ver.4.2〕

```
解放するﾓｼﾞｭｰﾙの選択:

RRR-73.BAS
RRR-71.BAS
RRR-72.BAS
```

④ rrr-72.BAS がなくなります. これは f·2 キーで確認できます.

```
C/ﾌﾟﾛｸﾞﾗﾑ選択:

RRR-73.BAS
RRR-71.BAS
plus
```

⑤ 同様にファイルメニューから U を行い, ファイル名に rrr-71.BAS を指定して ⏎ します. rrr-71.BAS がメモリから消えます. rrr-73.BAS のみ残ります.

【例題 31】 メインプログラムの設定

次の 3 つのプログラムを各々rrr-74, rrr-75, rrr-76 で保存し, 次に各々をメモリに呼んで, rrr-75, rrr-74, rrr-76 の順で実行せよ.

RRR-74
```
INPUT x%
PRINT x%
END
```

RRR-75
```
INPUT x%
PRINT x% * x%
END
```

RRR-76
```
INPUT x%
PRINT x% * x% * x%
END
```

【解　説】 ◎ 複数プログラムを読み込んで任意のプログラムを実行します.

① rrr-74, rrr-75, rrr-76 のプログラムをつくり各々保存します.

② ファイルメニューを選択します.

③ "L/サブファイル読込 ..."(バージョン 4.5) または L/サブファイル常駐 ..."(バージョン 4.2) を選びます. ファイル名を問合わせてきますので, rrr-74 ⏎ とします. このとき, 形式は M/モジュールとします. ファイル rrr-74 がメモリに読み込まれます.

④ 同様に rrr-75 と rrr-76 をメモリに読み込みます.

⑤ f·2 をおすと読み込まれたファイル名がリストされます.

```
C/ﾌﾟﾛｸﾞﾗﾑ選択:

RRR-74.BAS
RRR-75.BAS
RRR-76.BAS
```

⑥ まず rrr-75 を実行します.

実行メニューを選びます.

"M/メインモジュールの設定 ... "を選びます.

⑦ "rrr-75.BAS"を選択して ↵ します. rrr-75 がメインプログラムとなります.

⑧ f·5 で rrr-75 が実行されます.

⑨ 同様に実行メニューから M/メインモジュールの設定 ... "を選び, rrr-76 ↵ により rrr-76 をメインプログラムとして f·5 で rrr-76 を実行します.

⑩ 同様に実行メニューから"M/メインモジュールの設定 ... "を選び, rrr-74 ↵ として f·5 で rrr-74 を実行します.

【結 果】

```
? 3
 9
? 4
 64
? 8
 8
```

【例題 32】 マルチモジュールプログラムの保存

【例題 31】でつくった RRR-74, RRR-75, RRR-76 の各プログラムを一緒に保存せよ.

【解 説】 ◎ マルチモジュールプログラムを保存します.

① RRR-74, RRR-75, RRR-76 の 3 つのプログラムを同時に保存することができます.

② サブファイル常駐により 3 つのプログラムがメモリにあるものとします. ファイルメニューを選択します.

"V/すべて保存"を選びます. 3 つのプログラムが同時に保存されます.

③ ファイル名はメインモジュール (今回は rrr-74.BAS) の rrr-74 拡張子を.MAK とするメインファイルをつくります.

【例題 33】 マルチモジュールプログラムの読み込み

【例題 32】で保存したマルチモジュールプログラムを読み込め.

【解 説】 ◎ マルチモジュールプログラムを読み込みます.

① ファイルメニューを選択します.

② "O/読込 ... "を選びます.

③ ファイル名に RRR-74.MAK を入力します. 3 つのモジュールからなるマルチプログラムが読み込まれます.

④ 確認のため f·2 キーをおすと, 次のように 3 つのプログラム名がかかれています.

```
RRR-74.BAS
RRR-75.BAS
RRR-76.BAS
```

【例題 34】　ドキュメントファイルの作成

次のドキュメントをつくりファイルワープロ１．として保存せよ．

```
拝啓
        ますますご清栄のこととお喜び申し上げます

        さて，下記の通り定例会議を開催しますので
ご出席ください

                                          敬具
                    記
１．日時：
２．場所：
３．議題：
４．出席者：
５．その他：


                 以上
```

【解　説】　◎　ドキュメントファイルをつくります．

① ファイルメニューを選びます．

② ”C/サブファイル作成 ...”を選びます．

③ 次のファイル名入力画面となります．

〔Ver.4.5〕

```
            C/サブファイル作成
 ファイル名，ファイル形式を設定してください

 N/ファイル名：

        (*) M/モジュールプログラム
        ( ) I/インクルードファイル
        ( ) D/ドキュメントファイル

     < 確認 >  < 取消 >  < H/ヘルプ >
```

〔Ver.4.2〕

```
     N/ﾌｧｲﾙ名：

        形式
   (*) M/ﾓｼﾞｭｰﾙ
   ( ) I/ｲﾝｸﾙｰﾄﾞ
   ( ) D/ﾄﾞｷｭﾒﾝﾄ
```

④ ファイル名を”ワープロ１．”，”D/ドキュメント”を選択します．通常のプログラムはM/モジュールによって，構文チェックが行われます．ドキュメントファイルではBASIC文を使うわけではありませんからD/ドキュメントによって，構文チェックをしないようにします．

〔Ver.4.5〕

```
C/サブファイル作成
ファイル名，ファイル形式を設定してください

N/ファイル名：  ワープロ1.

( ) M/モジュールプログラム
( ) I/インクルードファイル
(*) D/ドキュメントファイル
```

〔Ver.4.2〕

```
N/ファイル名：     ワープロ1

形式
( ) M/モジュール
( ) I/インクルード
(*) D/ドキュメント
```

⑤　文章をかき込みます.

⑥　ファイルメニューから P で印刷，ファイルメニューから S でファイルに保存されます.

⑦　この機能を使って CONFIG.SYS や AUTOEXEC.BAT などを書き換えることができます.

【例題35】　ドキュメントファイル

【例題 34】のドキュメントファイルを使って，次のドキュメントをつくれ.

```
拝啓
        ますますご清栄のこととお喜び申し上げます

        さて、下記の通り定例会議を開催しますので
ご出席ください

                                        敬具
                     記
1．日時：     11月15日 15時から16時
2．場所：     第三会議室
3．議題：     創立30周年行事について
4．出席者：   総務部全員
5．その他：   前回の宿題を持参のこと


                   以上
```

【解　説】　◎　ドキュメントファイルの追加をします.

①　ファイルメニューから O ，ワープロ1. ↵ により，【例題 34】のドキュメントを読み出します.

②　カーソルを移して，"11 月 15 日……持参のこと"を追加して記入します.

③　保存はファイルメニューから A ，ワープロ2. ↵ ，印刷はファイルメニューから P です.

【例題36】　インクルードファイルの作成

下のインクルードファイルをファイル名"HAIRETSU.IN"でつくれ.

```
DIM a$(20)
COMMON a$()
```

【解　説】 ◎ インクルードファイルの作成

① インクルードファイルをつくります．この使い方はII編24章で扱います．

② ファイルメニューを選びます．次に"C/サブファイル作成 ..."を選びます．

③ ファイル名を問合わせますので"hairetsu.in"とします．形式を"I/インクルード"とします．↵により初期画面へもどります．

〔Ver.4.5〕

C/サブファイル作成
ファイル名，ファイル形式を設定してください

N/ファイル名： hairetsu.in

( ) M/モジュールプログラム
(*) I/インクルードファイル
( ) D/ドキュメントファイル

< 確認 > < 取消 > < H/ヘルプ >

〔Ver.4.2〕

N/ﾌｧｲﾙ名：　hairetsu.bas

形式
( ) M/ﾓｼﾞｭｰﾙ
(*) I/ｲﾝｸﾙｰﾄﾞ
( ) D/ﾄﾞｷｭﾒﾝﾄ

確認

取消

④ 問題の2行をかきます．これは配列A$(20)を宣言し，これをグローバル変数とするものです．

⑤ ファイルメニューを選択し，Aにより保存を選びます．ファイル名をb:hairetsu.in，形式を"T/テキスト"として↵します．インクルードファイルがドライブbに格納されます．

【例題37】 インクルードファイルの実行と表示

次のプログラムをファイル名RRR-10.BASでつくり，次に【例題36】でつくったインクルードファイルをつないで実行し，次にインクルードファイルを表示せよ．

```
REM $INCLUDE: 'B:HAIRETSU.IN'
FOR i = 0 TO 19
    READ a$(i)
NEXT i
FOR i = 0 TO 19
    IF a$(i) = "Tokyo" THEN PRINT a$(i)
NEXT i
END
DATA Osaka,Nagoya,Ueno,Kyoto
DATA Kobe,Nara,Kagoshima,Sapporo
DATA Yamaguchi,Yokohama,Chiba,Tsu
DATA Fukushima,Tokyo,Sendai,Miyagi
DATA Shizuoka,Hamamatsu,Mishima,Kanazawa
```

【解　説】 ◎ インクルードファイルの連結と表示をします．

① 【例題36】のファイルHAIRETSU.INを読み込みます．ファイルメニューからLで次の画面となります．ファイル名をB：HAIRETSU.IN，I/インクルードファイルを指定して↵します．

これでインクルードファイルが読み込まれ，ファイル RRR-10 につながります.

```
                                 RRR-10.BAS
                              L/サブファイル読込
ファイル名，ファイル形式を設定してください

N/ファイル名:  b:hairetsu.in

B:¥
                        F/ファイル                          D/ディレクトリ

  RENSHU-1.BAS   RRR-51.BAS    RRR-73.BAS          [-A-]
  RENSHU-2.BAS   RRR-6.BAS     RRR-74.BAS          [-B-]
  RRR-10.BAS     RRR-7.BAS     RRR-75.BAS
  RRR-100.BAS    RRR-70.BAS    RRR-76.BAS
  RRR-2.BAS      RRR-71.BAS    RRR-85.BAS
  RRR-50.BAS     RRR-72.BAS    RRR-95.BAS

      ( ) M/モジュールプログラム
      (*) I/インクルードファイル
      ( ) D/ドキュメントファイル

                                        < 確認 >  < 取消 >  < H/ヘルプ >
```

② f･5 で実行します．Tokyo と表示されます.

③ 表示メニューから"L/インクルードファイル表示"を選択します．RRR-10 で使われているインクルードファイル hairetsu.in が表示されます.

④ 次のようにインクルードファイルの実行される部分へ挿入された形で表示されます.

```
                                                RRR-10.BAS
REM $INCLUDE: 'B:HAIRETSU.IN'
DIM a$(20)
COMMON a$()

FOR i = 0 TO 19
    READ a$(i)
NEXT i
FOR i = 0 TO 19
    IF a$(i) = "Tokyo" THEN PRINT a$(i)
NEXT i
END
DATA Osaka,Nagoya,Ueno,Kyoto
DATA Kobe,Nara,Kagoshima,Sapporo
DATA Yamaguchi,Yokohama,Chiba,Tsu
DATA Fukushima,Tokyo,Sendai,Miyagi
DATA Shizuoka,Hamamatsu,Mishima,Kanazawa
```

⑤ 項③を再度行うと元へもどります.

【例題 38】　インクルードファイルの編集

【例題 36】のインクルードファイル"hairetsu.in"の修正画面をつくれ.

【解　説】　◎　インクルードファイルの編集をします.

① 表示メニューから"I/インクルードファイル編集"を選びます.

② インクルードファイル"hairetsu.in"が表示され，編集が可能となります.

```
DIM a$(20)
COMMON a$()
```

③ 編集後はファイルメニューを選択し，[S] によって保存を選び保存します.

6章 表　　示

【例題39】　画面分割

次のプログラムを使い，画面を分割して，上をメインルーチン，下をPRのサブルーチンの表示とせよ．なおこのプログラムは1から100の和を求めるものとする．

```
DECLARE SUB pr (z)
t = 0
FOR i = 1 TO 100
    t = t + i
NEXT i
CALL pr(t)
END


SUB pr (z)
    PRINT "1+2+3+････+100="; z
END SUB
```

【結　果】

```
1+2+3+････+100= 5050
```

【解　説】　◎　画面を上下に分割します．

① メインルーチンとサブルーチンPRをつくります．ファイル名RRR-70で保存します．

② 表示メニューを選びます．ここで，"P/画面分割"を選びます．分割をもどすときも，この操作です．

③ 次の2画面となります．上下ともメインルーチンです．

RRR-70.BAS

```
DECLARE SUB pr (z)
t = 0
FOR i = 1 TO 100
    t = t + i
NEXT i
CALL pr(t)
END
```

RRR-70.BAS

```
DECLARE SUB pr (z)
t = 0
FOR i = 1 TO 100
    t = t + i
NEXT i
CALL pr(t)
END
```

ﾀﾞｲﾚｸﾄ ﾓｰﾄﾞ

④　f･6 によって下側がアクティブウィンドウとなります．SHIFT と f･2 をおすと内容はサブルーチン PR となります．

RRR-70.BAS

```
DECLARE SUB pr (z)
t = 0
FOR i = 1 TO 100
    t = t + i
NEXT i
CALL pr(t)
END
```

RRR-70.BAS:pr

```
SUB pr (z)
    PRINT "1+2+3+････+100="; z
END SUB
```

ﾀﾞｲﾚｸﾄ ﾓｰﾄﾞ

⑤　さらに f･6 をおすとダイレクトモード，さらに f･6 をおすと上側がアクティブウインドウとなります．

【例題 40】　アクティブ画面のフルスクリーン

【例題 39】のプログラムでビューウィンドウを 2 分割し，上段にメインプログラム，下段にサブルーチン PR を表示し，下段をアクティブ画面としてからその画面をフルスクリーンとせよ．

【解　説】　◎　アクティブ画面をフルスクリーンとします．

①　【例題 39】の手順で，ビューウィンドウを 2 分割し，下段をアクティブ画面とし，かつサブルーチンを表示する画面とします．

②　CTRL + f･10 によりアクティブ画面をフルスクリーンとします．この操作で元へもどります．

RRR-70.BAS:pr

```
SUB pr (z)
    PRINT "1+2+3+････+100="; z
END SUB
```

【例題 41】　アクティブ画面の拡大・縮小

【例題 39】のプログラムを表示した後，ダイレクトモードウィンドウを拡大せよ．

【解　説】　◎　アクティブ画面の拡大・縮小をします．

①　【例題 39】のプログラムをつくります．

```
                                        RRR-70.BAS
DECLARE SUB pr (z)
t = 0
FOR i = 1 TO 100
    t = t + i
NEXT i
CALL pr(t)
END




                                        ﾀﾞｲﾚｸﾄ ﾓｰﾄﾞ
```

② [f･6] キーによって，ダイレクトモードウィンドウをアクティブ画面とします．

③ [GRPH] キーと [+] キーをおすとそのたびにダイレクトモードウィンドウが拡大します．

```
                                        RRR-70.BAS
DECLARE SUB pr (z)
t = 0
FOR i = 1 TO 100
    t = t + i
NEXT i
CALL pr(t)
END


                                        ﾀﾞｲﾚｸﾄ ﾓｰﾄﾞ
```

④ [GRPH] キーと [−] キーによって，逆にアクティブウィンドウを縮小できます．

〔ウィンドウの分割のまとめ〕

ウィンドウは①ビュー・ウィンドウ，②ダイレクト・モード・ウィンドウ，③ウォッチ・ウィンドウの３種があります．

〔Ver.4.5〕

```
 F/ﾌｧｲﾙ  E/編集  V/表示  S/検索  R/実行  D/ﾃﾞﾊﾞｯｸﾞ  C/関数  O/ｵﾌﾟｼｮﾝ  H/ﾍﾙﾌﾟ
 RRR-70.BAS t: <FALSE>
 RRR-70.BAS i: <FALSE>
                                  RRR-70.BAS
DECLARE SUB pr (z!)
t = 0
FOR i = 1 TO 100
    t = t + i
NEXT i
CALL pr(t)
END



                                  ﾀﾞｲﾚｸﾄ ﾓｰﾄﾞ


<SHIFT+F1=ﾍﾙﾌﾟ> <F6=窓切換> <F2=SUB一覧> <F5=実行> <F8=ﾄﾚｰｽ> |      00001:001
```

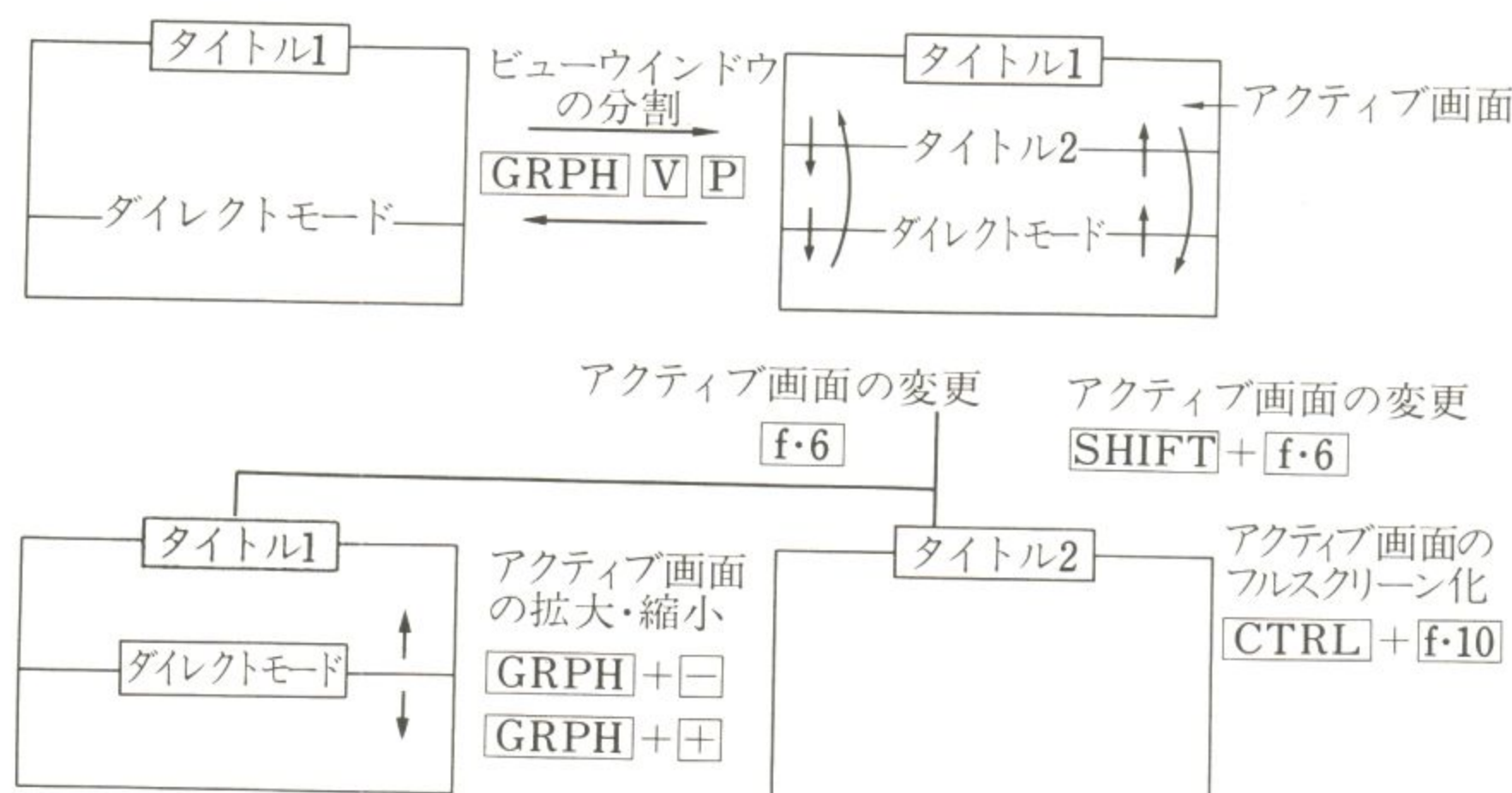
ウォッチ・ウインドウ

ビュー・ウインドウ

ダイレクト・モード・ウインドウ

ビュー・ウィンドウではタイトルが反転表示されている画面がアクティブ画面です．ダイレクト・モード・ウィンドウがアクティブな場合は”ダイレクト・モード”の部分が反転表示されます．

ビュー・ウィンドウは２つに分割できます．分割されたビュー・ウィンドウのどちらかをアクティブにしたり，ダイレクト・モードをアクティブにしたりできます．また，アクティブな画面をフルスクリーンにできます．また，アクティブウィンドウの大きさを拡大したり，縮小したりできます．これらの関係を図に示します．

タイトル1
ダイレクトモード
ビューウインドウの分割
GRPH V P
タイトル1
タイトル2
ダイレクトモード
アクティブ画面
アクティブ画面の変更 f・6
アクティブ画面の変更 SHIFT＋f・6
タイトル1
ダイレクトモード
アクティブ画面の拡大・縮小
GRPH＋－
GRPH＋＋
タイトル2
アクティブ画面のフルスクリーン化
CTRL＋f・10

【例題 42】　サブルーチンの一覧

次のプログラムをつくり，サブルーチンの一覧を確認せよ．ただし，このプログラムはファイル名は RRR-7.BAS とします．

```
DECLARE SUB a1 (z%, zz%, zzz%)
DECLARE SUB a2 (z AS INTEGER, zz AS INTEGER)
INPUT x%
STOP
CALL a1(x%, xx%, xxx%)
PRINT xx%
PRINT xxx%
END


SUB a1 (z%, zz%, zzz%)
zz% = 0
FOR i = 1 TO z%
    zz% = zz% + i
NEXT i
STOP
CALL a2(z%, yy%)
zzz% = yy%
END SUB

SUB a2 (y%, yy%)
z% = 1
FOR i = y% TO 1 STEP -1
    z% = z% * i
NEXT i
yy% = z%
STOP
END SUB
```

【解　説】　◎　サブルーチンの一覧をつくります．

①　表示メニューを選びます．

② ここで"S/SUB 一覧 ..."を選びます. 次のようにサブルーチンの一覧ができます.

〔Ver.4.5〕

```
                V/表示
                                  S/SUB一覧
C/編集するプロシージャ，ウィンドウを選択してください

  RRR-7.BAS
    a1
    a2

RRR-7.BAS:=メインモジュール
```

〔Ver.4.2〕

```
                  V/表示

C/ﾌﾟﾛｸﾞﾗﾑ選択:

  RRR-7.BAS
    a1
    a2

RRR-7.BAS:=ﾒｲﾝﾌﾟﾛｸﾞﾗﾑ
```

③ 矢印キー [↑] または [↓] でプログラムを選択できます. たとえば"a1"を選んで [↵] するとサブルーチン a1 が表示されます.

```
                                 RRR-7.BAS:a1
SUB a1 (z%, zz%, zzz%)
zz% = 0
FOR i = 1 TO z%
    zz% = zz% + i
NEXT i
STOP
CALL a2(z%, yy%)
zzz% = yy%
END SUB
```

④ なお [f・2] によっても同様の一覧が選べます.

【例題 43】　サブルーチンの選択

【例題 42】のプログラムを使って, メインルーチン, サブルーチンを順に呼び出せ.

【解　説】 ◎ メインプログラム，サブルーチンを順に呼び出します．

① 表示メニューを選びます．

② ここで"E/次の SUB"を選びます．

③ 現在表示されている次のサブルーチンまたはファンクションが表示されます．たとえば，a1 のサブルーチンが表示されている状態で，この手順を行うとサブルーチン a2 が表示されます．

④ なお，SHIFT ＋ f･2 で同じことができます．このくり返しで，メインプログラム，サブルーチン，ファンクションを呼び出します．

7章 編　　集

【例題44】　編集のとり消し

次のプログラムをつくり，次にⒷのように1行目を修正している途中でこれをとり消して，元へもどせ.

```
PRINT "Quick Basic ﾊ ﾍﾞﾝﾘﾅ ｹﾞﾝｺﾞﾃﾞｽ｡"
END
```

Ⓑ
```
PRINT "Quick C ｹﾞﾝｺﾞﾃﾞｽ｡"
END
```

【解　説】 ◎ アンドウ機能で修正をとり消します.

① 問題のプログラムをつくります.

② 次にカーソルを"B"におき DEL キーをおして文字を消し，C をおしてⒷのようにします.

③ ここで編集メニューを選択し，"U/元に戻す"を選びます.
最初の状態にもどります.

【例題45】　編集機能によるコピーとペースト

項③のプログラムを使い，ENDの前に1行目から3行目までを挿入して，数x，yを入力して積を求めて表示し，次に再度xとyを入力して積を求めて表示するプログラムとせよ.

【解　説】 ◎ E/編集のC/コピーとP/ペーストを使います.

① 1文字単位の挿入，削除，訂正は DEL，BS，INS キーなどで簡単に行えます．しかし，複数行を扱うときはE/編集機能を使うと簡単です.

② 今回はINPUT x，yとz=x*yとPRINT x;"*";y;"=";zの3行をENDの前へ挿入します.

③ 第1にどの行を扱うかを指定します．以下のプログラムを使い，ENDのEにカーソルをおき，CTRL と SHIFT と E キーを同時におします．第3行のPRINT以下が反転表示されます．さらに CTRL と SHIFT と E キーをおすと1つ上の第2行が反転表示されます．もう一度これを行います．第1行も反転表示します．この反転表示部分が挿入，削除の対象です．なお矢印キーをおすと元へもどります.

```
INPUT x, y
z = x * y
PRINT x; "*"; y; "="; z
END
```

Eにカーソルをおき，CTRL と SHIFT をおしたまま E キーを3回おすと第1～第3行が反転表示される.

④ 次に E/編集モードを選びます．

⑤ ここではC/コピーを選びます．画面上は変化はありませんが，指定された第1～第3行がクリップボードと呼ばれるメモリにかき込まれます．

⑥ カーソルを END の E の上におきます．編集メニューを選びます．今度は P/ペーストを選びます．

⑦ GRPH + BS キーをおすと元へもどります．

⑧ クリップボードの3行が挿入されます．

```
INPUT x, y
z = x * y
PRINT x; "*"; y; "="; z
INPUT x, y
z = x * y
PRINT x; "*"; y; "="; z
END
```

3行挿入

⑨ クリップボードの3行は残っていますからいつでもどこへでも挿入することができます．たとえば，サブルーチンの画面を出して，サブルーチンの中の指定行へ挿入することも可能です．

【実行結果】

```
? 2,3
 2 * 3 = 6
? 5,9
 5 * 9 = 45
```

【例題 46】 編集機能を使った削除

【例題 45】のプログラムの4行目の INPUT x, y を削除し，次に新しい4行目の z=x*y を z=x/y とし，新しい5行目の*を/に変更せよ．このプログラムは x と y を入力して積を求めて表示し，次に x/y を z に求めて表示するプログラムである．

【解 説】 ◎ E/編集機能を使って行単位の削除をします．

① E/編集の T/カットと E/削除は行単位で削除します．

② まず削除する行を指定します．今回は4行目の INPUT x, y を削除しますから，5行目の先頭の z にカーソルをおき CTRL + SHIFT + E キーをおします．4行目が反転表示されます．

③ 編集メニューを選びます．T/カットまたは E/削除を選びます．4行目の INPUT x, y が削除されます．T/カットの場合は第4行がクリップボードに残ります．E/削除ではクリップボードにも残りません．

④ 次のようになります．

```
INPUT x, y
z = x * y
PRINT x; "*"; y; "="; z
z = x * y
PRINT x; "*"; y; "="; z
END
```

⑤ 4行目の*を/に，5行目の*を/に変更します．

```
INPUT x, y
z = x * y
PRINT x; "*"; y; "="; z
z = x / y
PRINT x; "/"; y; "="; z
END
```

【実行結果】

```
 8,5                    8と5を入力の例
8 * 5 = 40
8 / 5 = 1.6
```

【例題 47】　サブルーチンへ行単位でコピー

数 x を入力し，x^2 を表示し，次に，サブルーチン a へ移って，x^3 を表示するプログラムをつくれ．ただし，サブルーチンでは，メインの INPUT 文に PRINT 文をコピーして1部修正して使うものとする．

【解　説】　◎　メインプログラムの行をサブルーチンへコピーします．

① 次のようにメインプログラムとサブルーチンをつくります．

```
INPUT x
y = x * x
PRINT x; "^ 2 ="; y
CALL a(x)
END
```

END の後に SUB a(x)をかき [↵] またはカーソルの行をかえると下の画面となります．

```
SUB a (x)

END SUB
```

② メインプログラムの第2行と第3行をサブルーチンへ挿入します．したがって，まずメインプログラムの第3行目の CALL a(x)の C にカーソルをおき，[CTRL] + [SHIFT] + [E] を2回おします．2行～3行が反転表示されます．

③ E/編集メニューを選び，C/コピーを選んで，2行をクリップボードにかき込みます．

④ [SHIFT] + [f・2] キーで SUB a(x)のサブルーチンがアクティブ画面となります．

⑤ カーソルをサブルーチンの空白行におき編集メニューから C/コピーを選びます．

⑥ 次のように2行が挿入されます．

```
SUB a (x)
y = x * x
PRINT x; "^ 2 ="; y

END SUB
```

⑦ 空白にカーソルをおき [DEL] で空白をとり，次に，第2行に＊x を追加し，第3行の2を3に変更します．

```
SUB a
y = x * x * x
PRINT x; "^ 3 ="; y
END SUB
```

【実行結果】

```
? 5                              5を入力の例
 5 ^ 2 = 25
 5 ^ 3 = 125
```

8章 検　　索

【例題48】 検　索

次のプログラムを作成し，"i"を検索せよ．ただし，このプログラムは1から10までの各値の和，2乗の和，3乗の和を求めるものです．

```
FOR i = 1 TO 10
    x = x + i
    y = y + i * i
    z = z + i * i * i
NEXT i
PRINT x
PRINT y
PRINT z
END
```

【解　説】 ◎ 文字の検索をします．

① 文字，文字列の検索を行います．この例ではプログラムが短いため，簡単に1目でわかりますが，長いプログラムの中で特定の文字や文字列をさがすのに便利です．

② 検索メニューを選択します．

③ "F/検索"...を選びます．次の検索文字列指定画面となります．

〔**Ver.4.5**〕

```
                F/検索
検索する文字列と検索条件を設定してください

F/検索対象:

                               範囲
[ ] M/大小文字区別         ( ) 1. 現在のウィンドウ
[ ] W/全体一致             (*) 2. 現在のモジュール
                           ( ) 3. 全モジュール

        < 確認 >  < 取消 >  < H/ヘルプ >
```

〔**Ver.4.2**〕

```
F/検索文字列

        検索                      [ ] M/大小文字区別
  ( ) 1/アクティブウィンドウ      [ ] W/全体一致
  (*) 2/カレントウィンドウ
  ( ) 3/全体
                                      確認    取消
```

④ 今回は"i"で検索をしますので"i"をかき込みます．

⑤ ⏎ によって，1行目のFOR i=1　TO　10のiにカーソルが移ります．検索メニューで"R/次

を検索”を選ぶと次の i にカーソルが移ります.

⑥　なお，”M/大小文字区別”を選択していれば，i は小文字ですから PRINT x の I は検索しません. [TAB] キーで ”M/大小文字区別”の前の[]にカーソルをおき，スペースをおすと[X]となり，大文字・小文字の区別をします．再度スペースをおすと解除されます.

【例題 49】　文字列の検索

【例題 48】のプログラムを使って，PRINT を検索せよ.

【解　説】　◎　文字列を検索します.

①　【例題 48】の手順で検索文字列として”PRINT”を指定します. [↵] によって最初の PRINT にカーソルが移ります.

②　検索メニューを選び，次に”R/次を検索”を選びます．次の”PRINT”にカーソルを移します．以下同様に [f·3] をおすと順に次の”PRINT”を検索します.

【例題 50】　置　換

【例題 48】のプログラムを使って，PRINT を LPRINT に置きかえよ.

【解　説】　◎　文字列を検索し，その文字列を指定した文字列に置きかえます.

①　検索メニューを選びます.

②　”C/置換...”を選びます.

③　”F/置換前”に”print”[TAB]，T/置換後に”lprint”[TAB] をします．次に [TAB] キーで C/一括置換を選んで [↵] または [C] とします.

〔Ver.4.5〕

```
                    C/置換
置換する文字列と置換条件を設定してください

F/置換前： PRINT

T/置換後： LPRINT

                              範囲
[ ] M/大小文字区別      ( ) 1. 現在のウィンドウ
[ ] W/全体一致          (*) 2. 現在のモジュール
                        ( ) 3. 全モジュール

              < V/逐次確認 >  < C/一括置換 >
                  < 取消 >  < H/ヘルプ >
```

〔Ver.4.2〕

```
F/置換前        print

T/置換後        lprint

              置換
   ( ) 1/アクティブウィンドウ          [ ] M/大小文字区別
   (*) 2/カレントウィンドウ            [ ] W/全体一致
   ( ) 3/全体

         V/逐次確認          C/一括置換          取消
```

④ プログラム中のPRINTはすべてLPRINTに置換されます.

```
FOR i = 1 TO 10
    x = x + i
    y = y + i * i
    z = z + i * i * i
NEXT i
LPRINT x
LPRINT y
LPRINT z
END
```

【例題51】 部分置換

【例題50】でつくったプログラムのうちLPRINT yのみPRINT yにかえよ.

【解　説】 ◎ 検索文字列群の1部のみを置換します.

① 【例題50】でつくったプログラムを使います.

② 検索メニューを選び,次に"C/置換"を選択します.

③ "F/置換前"として"lprint","T/置換後"として"print"を指定します.

④ 次にV/逐次確認を選んで[↵] または [V] とします.カーソルがPRINT xにおかれて次の画面となります.

〔Ver.4.5〕

```
 F/ファイル  E/編集  V/表示  S/検索  R/実行  D/デバッグ  C/関数  O/オプション  H/ヘルプ
                                  Untitled
FOR i = 1 TO 10
    x = x + i
    y = y + i * i
    z = z + i * i * i
NEXT i
LPRINT x
LPRINT y
LPRINT z
END

                                   C/置換
          処理を選択してください

          < C/置換 >  < S/スキップ >  < 取消 >  < H/ヘルプ >
```

〔Ver.4.2〕

```
  F/ファイル  E/編集  V/表示  S/検索  R/実行  D/デバッグ  C/関数    H/ヘルプ (f・1)
                                   RRR-4.BAS
FOR i = 1 TO 10
    x = x + i
    y = y + i * i
    z = z + i * i * i
NEXT i
LPRINT x
LPRINT y
LPRINT z
END

                    C/置換する      S/置換しない      取消
```

⑤　LPRINT x は置換しませんので"S/スキップ"(バージョン 4.5), "S/置換しない" (バージョン 4.2) を選んで[↵] または [S] とします. 次の LPRINT y へ進みます.

⑥　カーソルが"LPRINT y"におかれます.

⑦　ここでは LPRINT y を PRINT y に置換しますから"C/置換" (バージョン 4.5), "C/置換する" (バージョン 4.2) にカーソルを移し[↵] または [C] とします. "LPRINT y"が"PRINT y"に置換されます.

```
FOR i = 1 TO 10
    x = x + i
    y = y + i * i
    z = z + i * i * i
NEXT i
LPRINT x
PRINT y
LPRINT z
END
```

【例題 52】　ラベルの検索

次のプログラムで検索文字列"ex"を指定して，ラベル"ex:"を検索せよ．ただし，このプログラムは 1 から 10 までの和，2 乗の和，3 乗の和を求めて表示するものであるが，3 乗の和が 200 以上となったら結果を表示するものです．

```
FOR i = 1 TO 10
    x = x + i
    y = y + i * i
    z = z + i * i * i
    IF z >= 200 THEN GOTO ex
NEXT i
ex:
PRINT x
PRINT y
PRINT z
END
```

【解　説】 ◎ ラベルの検索を行います.

① 検索メニューを選択します.

② 続いて"L/ラベル検索（バージョン4.5），"L/ラベル ..."（バージョン4.2）を選びます.

③ ラベル名としてex:をかき⏎します. ラベル"ex:"を検索して，カーソルを移します. 次のラベル入力画面となります.

〔Ver.4.5〕

```
                    L/ラベル検索
検索する文字列と検索条件を設定してください

F/検索対象： ex:

                                  範囲
[ ] M/大小文字区別          ( ) 1. 現在のウィンドウ
[ ] W/全体一致              (*) 2. 現在のモジュール
                            ( ) 3. 全モジュール

              < 確認 >  < 取消 >  < H/ヘルプ >
```

〔Ver.4.2〕

```
F/検索文字列   ex:

        検索                    [ ] M/大小文字区別
  ( ) 1/アクティブウィンドウ      [ ] W/全体一致
  (*) 2/カレントウィンドウ
  ( ) 3/全体
                                     確認      取消
```

9章　デバッグ・関数

【例題 53】　トレース

次のプログラムは seisu%に整数を入力し，階乗を求めるプログラムである．これを１ステップずつトレースせよ．

```
DECLARE FUNCTION kaijou (s%)
INPUT seisu%
PRINT seisu%; "!="; kaijou(seisu%)
END


FUNCTION kaijou (s%)
    IF s% = 0 THEN
        kaijou = 1
    ELSE
        kaijou = s% * kaijou(s% - 1)
    END IF
END FUNCTION
```

【解　説】　◎　トレースを行います．

① トレースとは１ステップずつ実行することです．プログラムの流れを確認できます．【例題 55】で行う，整数のウォッチと合わせて行うことでプログラムのデバッグを簡単に行います．

② トレース機能は [f･8] キーと [f･10] キーです．[f･8] キーはサブルーチンやファンクションの中までトレースするのに対し，[f･10] はメインプログラムの中だけトレースします．

③ [f･8] キーまたは [f･10] キーをおすたびに実行されているステートメントが緑色になります．また途中で [f･5] キーをおすと通常の実行に移ります．途中でトレースをやめて高速で終了させる場合に使います．

④ 実際にトレースを行います．

まず最初の [f･8] をおします．"INPUT seisu%"が緑色となります．

次に再び [f･8] をおします．

?が表示されます．整数の入力です．今回は 6[↵] とします．

"PRINT seisu%;"!=";kaijou(seisu%)"が緑となります．続いて [f･8] をおします．

FUNCTION kaijou(s%)の中のトレースに入り"IF s%=0　THEN"が緑色となります．

続いて [f･8] をおします．

"kaijou=s%*kaijou(s%-1)"が緑色となります．

以下 [f･8] をおすたびに"IF s%=0　THEN"と"kaijou=s%*kaijou(s%-1)"をくり返し緑色とします．s%が０となると次の [f･8] で kaijou=1 が緑色となります．

[f･8] をおします．

"ELSE"が緑色となります．

[f･8] をおします.

"END FUNCTION"が緑色となります.

[f･8] をおします.

"END IF"が緑色になります.

次からの [f･8] によって"END FUNCTION"と"END IF"をくり返した後ファンクションからぬけ出てメインプログラムへもどります.

"END"が緑色となります.

[f･8] をおします. 全体が白くなって終了です.

⑤ [f･10] を使うと, 整数を入力した後, ファンクションは自動実行して, すぐ "PRINT seisu%;"!=";kaijou(seisu%)"が緑色となり, 次の [f･10] で"END"が緑色となって, 次の [f･10] で終了です.

【例題 54】 低速トレース

【例題 53】のプログラムを使って, 低速でトレースを自動実行せよ.

【解 説】 ◎ トレースを低速で実行します.

① トレースを低速で行います.

デバッグメニューを選択します.

② T/トレースを選びます. 元の画面にもどります.

③ [f･5] をおすと"INPUT seisu%"が緑色となり, 入力待ちとなりますので整数を入力します. たとえば 6 [↵] です.

④ あとは自動的に1ステップずつ進めていきます. 実行のステートメントが緑色にかわります.

⑤ なお, デバッグメニューを選ぶと"T/トレース"の前に*(バージョン 4.5), >(バージョン 4.2)がついていて, トレースが選ばれていることがわかります.

⑥ 解除はT/トレースを選んで [↵] します.

【例題 55】 ウォッチ

次のプログラムを1ステップずつ実行し, そのつど変数 i と x の値をチェックせよ. このプログラムは i を1から10までとし, x にその和を求めるものです.

```
FOR i = 1 TO 10
    x = x + i
NEXT i
PRINT x
END
```

【解 説】 ◎ ウォッチ機能で変数の値をチェックします.

① ウォッチ機能は指定した変数や式の値をチェックするものです.

② デバッグメニューを選択します.

③ "A/ウォッチ"を選びます. 次の画面となります.

〔**Ver.4.5**〕

```
                    A/ウォッチの追加
ウォッチウィンドウに追加する式を入力してください



          < 確認 > < 取消 > < H/ヘルプ >
```

〔**Ver.4.2**〕

```
ウォッチウィンドウに登録する式・変数の設定：
[                                          ]

          確認            取消
```

④　ウォッチすなわちチェックをする変数名 i をかきます.

⑤　[↵] をすると次図のようにウォッチする i が左上部に表示されます.

```
 RRR-2.BAS i:
                                   RRR-2.BAS
FOR i = 1 TO 10
    x = x + i
NEXT i
PRINT x
END
```

⑥　同様の手順，デバッグメニューから [A]，[X] [↵] によって，ウォッチをする第２の変数 x が左上部に表示されます.

```
 RRR-2.BAS i:
 RRR-2.BAS x:
                                   RRR-2.BAS
FOR i = 1 TO 10
    x = x + i
NEXT i
PRINT x
END
```

⑦　実行します. [f・8] をおします. １ステップずつ進みます. まず変数 i，x とも０ですから次図のように，i，x の値として０が表示されます. "FOR i=1　TO　10"が緑色になっています.

```
 RRR-2.BAS i:  0
 RRR-2.BAS x:  0
                                   RRR-2.BAS
FOR i = 1 TO 10
    x = x + i
NEXT i
PRINT x
END
```

⑧　続いて [f・8] をおします. i が１となります.

```
 RRR-2.BAS i:  1
 RRR-2.BAS x:  0
                                  RRR-2.BAS
FOR i = 1 TO 10
    x = x + i
NEXT i
PRINT x
END
```

⑨ [f･8] をおします．x=x+i により x が 1 となります．

```
 RRR-2.BAS i:  1
 RRR-2.BAS x:  1
                                  RRR-2.BAS
FOR i = 1 TO 10
    x = x + i
NEXT i
PRINT x
END
```

⑩ [f･8] をおします．"x=x+i"が緑色となり i が 2 となります．

```
 RRR-2.BAS i:  2
 RRR-2.BAS x:  1
                                  RRR-2.BAS
FOR i = 1 TO 10
    x = x + i
NEXT i
PRINT x
END
```

⑪ [f･8] をおします．"NEXT i"が緑色となり x は 1+2 の結果の 3 となります．

```
 RRR-2.BAS i:  2
 RRR-2.BAS x:  3
                                  RRR-2.BAS
FOR i = 1 TO 10
    x = x + i
NEXT i
PRINT x
END
```

⑫ このくり返しで i が 10，x が 55 となります．

```
 RRR-2.BAS i:  10
 RRR-2.BAS x:  55
                                  RRR-2.BAS
FOR i = 1 TO 10
    x = x + i
NEXT i
PRINT x
END
```

⑬ 次に [f･8] をおすと"PRINT x"が緑色となり i が 11，x が 55 となります．

```
 RRR-2.BAS i:  11
 RRR-2.BAS x:  55
                                  RRR-2.BAS
FOR i = 1 TO 10
    x = x + i
NEXT i
PRINT x
END
```

⑭ [f･8] をおすと"END"が緑色となり，さらに [f･8] をおすと"END"が白となって終了です．

⑮ バージョン 4.5 では I/簡易ウォッチが設けられました．ウォッチする式にカーソルをおき，デ

バッグメニューからI/簡易ウォッチを選ぶか，または SHIFT ＋ f･9 でウォッチ対象を決められます．

【例題 56】　ウォッチを設定してトレースの自動実行

【例題 55】のプログラムと i と x をウォッチに設定して自動トレースせよ．

【解　説】　◎　ウォッチを設けて自動トレースをします．

① 【例題 55】と同じ手順で i と x のウォッチを設定します．

② 次に【例題 53】の手順でトレースを設定します．

③ f･5 をおします．i と x の値を順に表示しながらトレースを実行していきます．

【例題 57】　ウォッチの削除

【例題 56】のように i と x をウォッチして自動トレースした後，i のウォッチを削除せよ．

【解　説】　◎　ウォッチの削除をします．

① 【例題 56】の手順で i と x をウォッチに設定して自動トレースします．

② デバッグメニューを選びます．

③ D/ウォッチの削除を選びます．次の画面となり，ウォッチを削除する変数名がリストアップされます．

〔**Ver.4.5**〕

```
RRR-2.BAS

D/ウォッチの削除
削除する式を選択してください

RRR-2.BAS i:
RRR-2.BAS x:

< 確認 >  < 取消 >  < H/ヘルプ >
```

〔**Ver.4.2**〕

```
RRR-2.BAS i:
RRR-2.BAS x:

確認    取消
```

④　矢印キーを使って RRR-2.BAS i:にカーソルを移し ⏎ します．i がウォッチから除かれます．

```
RRR-2.BAS x:
                                              RRR-2.BAS
FOR i = 1 TO 10
    x = x + i
NEXT i
PRINT x
END
```

> **【例題 58】　条件式を設定してトレース（ウォッチポイント）**
>
> 【例題 55】のプログラムを i=5 まで自動トレースで x をウォッチし，次に i=8 までを自動トレースでウォッチせよ．

【解　説】　◎　条件式を設定して，真になるまでトレースします．

①　条件式を設定して，式が真になるまでトレースします．これをウォッチポイントと呼びます．

②　【例題 55】の手順で x をウォッチに設定します．

③　【例題 53】の手順で自動トレースを設定します．

④　デバッグメニューを選び，”W/ウォッチポイント”を選びます．次の画面となります．

〔Ver.4.5〕

```
                    W/ウォッチポイント

     式を入力してください(式が真になるまで実行します)



                < 確認 >  < 取消 >  < H/ヘルプ >
```

〔Ver.4.2〕

```
   F/ファイル  E/編集  V/表示  S/検索  R/実行  D/デバッグ  C/関数    H/ヘルプ(f･1)
  RRR-2.BAS x:
                                          RRR-2.BAS
 FOR

 NEX  条件式の設定:(式=真になるまで実行します)
 PRI
 END


                    確認                    取消
```

⑤　条件は i=5 まで実行しますから i=5 と記入します．

⑥　⏎ します．条件式が左上に表示されます．現在は i=5 ではありませんから〈FALSE〉（偽）と表示されています．

```
 RRR-2.BAS x:
 RRR-2.BAS i = 5: <FALSE>
                                              RRR-2.BAS
FOR i = 1 TO 10
    x = x + i
NEXT i
PRINT x
END
```

⑦　これで準備は完了しましたので f・8 ，または f・10 または f・5 で実行します．今回は f・5 をおします．

トレースが行われ，i=5 となると画面左上の〈FALSE〉が〈TRUE〉にかわり i が 5 になったことを示します．このときの x の値は 10 です．また i=5 のときの x=x+i は行われていません．

```
 RRR-2.BAS x:  10
 RRR-2.BAS i = 5: <TRUE>
                                   RRR-2.BAS
FOR i = 1 TO 10
    x = x + i
NEXT i
PRINT x
END
```

⑧　続いて i=8 まで実行するものとします．デバッグメニューを選び，"W/ウォッチポイント"を選びます．i=8↵ とします．

```
 RRR-2.BAS x:  10                                       ↵ をしたところ
 RRR-2.BAS i = 5: <TRUE>
 RRR-2.BAS i = 8: <FALSE>
                                   RRR-2.BAS
FOR i = 1 TO 10
    x = x + i
NEXT i
PRINT x
END
```

⑨　f・5 でトレースを再開します．i=8 となると i=8 が〈TRUE〉となり停止します．x の値は 28 です．i=8 のときの x=x+i はまだ実行されていません．

```
 RRR-2.BAS x:  28
 RRR-2.BAS i = 5: <FALSE>
 RRR-2.BAS i = 8: <TRUE>
                                   RRR-2.BAS
FOR i = 1 TO 10
    x = x + i
NEXT i
PRINT x
END
```

⑩　f・5 を再度おします．実行が継続され x は 55，i=11 となって，終了します．ただし，i の値はウォッチしていませんから表示されません．

```
 RRR-2.BAS x:  55
 RRR-2.BAS i = 5: <FALSE>
 RRR-2.BAS i = 8: <FALSE>
                                   RRR-2.BAS
FOR i = 1 TO 10
    x = x + i
NEXT i
PRINT x
END
```

【例題 59】　ウォッチの削除

【例題 58】のウォッチを全部削除せよ．

【解　説】　◎　ウォッチを全部削除します．

①　【例題 55】の手順でウォッチを設け，自動トレースをします．

②　デバッグメニューを選びます．

③　”L/全ウォッチ削除”を選びます．全ウォッチが削除されます．

【例題60】　ブレークポイント

【例題55】のプログラムを x をウォッチして実行せよ．ただし，x=x+i のステートメントにブレークポイントをおき，そのステートメントを実行するとき停止するものとする．

【解　説】　◎　ブレークポイントを設けてプログラムを1度停止します．

①　ブレークポイントはプログラム中のステートメント行を指定して，その行を実行するとき一度プログラムを停止させます．変数をウォッチに設定すればプログラムが停止したところでの変数の値をチェックできます．

②　【例題55】の手順で変数 x をウォッチに設定します．

③　カーソルを x=x+i の行へおきます．

④　ここで [f･9] キーをおします．この行が赤くなります．これはブレークポイントが設定されたことを示します．これはデバッグメニューから”B/ブレークポイント”を選んでも同じです．

ブレークポイントはカーソルのある行に設けます．ブレークポイントは複数の行に設けることができますから，カーソルを移動して [f･9] をおすか，デバッグメニューから指定します．

⑤　[f･5] で実行します．x=x+i の行へくると一度停止します．その時の x の値をチェックすることができます．継続は [f･5] をおします．

```
 RRR-2.BAS x:  55
                                    RRR-2.BAS
FOR i = 1 TO 10
    x = x + i
NEXT i
PRINT x
END
```

最後は x の値が 55 となって終了します．

【例題61】　ブレークポイントの解除

【例題60】でつくったブレークポイントをすべて解除せよ．

【解　説】　◎　ブレークポイントをすべて解除します．

①　【例題60】のブレークポイントを利用します．この例ではブレークポイントは1つですが複数個あっても同じです．

②　デバッグメニューを選びます．”C/全ブレークポイントの削除”（バージョン 4.5），”C/全ブレークポイントの解放”（バージョン 4.2）を選びます．すべてのブレークポイントが解放されます．

③　1行の場合は，その行へカーソルをおき [f･9] をおします．

【例題 62】　ヒストリ

次のプログラムを実行し，終了したら，1 ステートメントずつ逆にさかのぼってプログラムの流れをチェックせよ．なお，このプログラムは乱数を 10 個つくって，0.3 以下のとき x1，0.3 をこえ 0.6 以下のとき x2，それ以外は x3 にその出現回数を数えるものです．

```
FOR i = 1 TO 10
    x = RND(5)
    IF x <= .3 THEN
        x1 = x1 + 1
    ELSE
        IF x <= .6 THEN
           x2 = x2 + 1
        ELSE
           x3 = x3 + 1
        END IF
    END IF
NEXT i
PRINT x1
PRINT x2
PRINT x3
END
```

【解　説】　◎　ヒストリは最大 20 までのステップの流れを記憶します．

① ヒストリは，最新の 20 ステップを記憶し，逆にさかのぼってステップを確認したり，さかのぼってから，また前へ進めることができます．

エラーがおきてプログラムが止まったとき，直前のステップを確認するのに大変便利です．

② デバッグメニューを選びます．

③ "H/ヒストリ"を選びます．元の画面にもどります．デバッグメニューを選ぶと"H/ヒストリ"の前に"*"(バージョン 4.5)，">"(バージョン 4.2) がついていて，ヒストリが選択されていることを示します．再度"H/ヒストリ"を選ぶとヒストリ機能は解除されます．

④ [f・5] でプログラムを実行します．プログラムは実行されて，"END"にカーソルがきます．ここで [SHIFT] キーと [f・8] キーをおすとプログラムのステップを逆にさかのぼっていきます．カーソルが"PRINT x3"→"PRINT x2"→"PRINT x1"…とさかのぼります．途中で [SHIFT] キーと [f・10] キーをおすと，元の流れでプログラムを進めていきます．

【例題 63】　次のステートメント

【例題 62】のプログラムを使って"x3=x3+1"にブレークポイントをおき実行し，次にカーソルを PRINT x1 において次のステートメントを設定してプログラムを継続させよ．

【解　説】　◎　次のステートメントを設定します．

① 次のステートメントは GOTO 文のようなもので，途中の実行を省略して，カーソルのあるステートメントへ実行を移します．

② カーソルを"x3=x3+1"におき [f・9] キーでブレークポイントをおきます．

③ [f・5] で実行します．x3=x3+1 の実行で止まります．x3=x3+1 はまだ実行していません．

④　カーソルを"PRINT x1"におきます.

⑤　デバッグメニューを選び，"S/次のステートメントの設定"（バージョン 4.5），"S/カレントステートメントの設定"（バージョン 4.2）を選びます．次のステートメントが"PRINT x1"となります.

⑥　f･5 をおします．x3=x3+1 を実行して，PRINT x1 に分岐し，PRINT x2，PRINT x3 を実行して終了します．結果は x3 のみ 1 です.

【結　果】

```
0        x１の値
0        x２の値
1        x３の値
```

> **【例題 64】　カーソルのある行まで実行**
>
> 【例題 62】のプログラムを使い，ウォッチに x1，x2，x3 を指定して，カーソルを x2=x2+1 におき，x2=x2+1 にプログラムの流れがくるまで実行することをくり返して，プログラムを実行せよ.

【解　説】　◎　f･7 キーはカーソルのある行までプログラムを実行します.

①　【例題 62】の要領で x1，x2，x3 をウォッチの対象とします．またカーソルを"x2=x2+1"の先頭の x におきます．次図となります.

```
RRR-3.BAS x1:
RRR-3.BAS x2:
RRR-3.BAS x3:
                                        RRR-3.BAS
FOR i = 1 TO 10
    x = RND(5)
    IF x <= .3 THEN
        x1 = x1 + 1
    ELSE
        IF x <= .6 THEN
           x2 = x2 + 1
        ELSE
           x3 = x3 + 1
        END IF
    END IF
NEXT i
PRINT x1
PRINT x2
PRINT x3
```

②　f･7 キーをおします．プログラムの流れが x2=x2+1 の実行にきたところで止まります．x2=x2+1 はまだ実行されていません．最初の乱数が 0.6 をこえているため x3=x3+1 を実行しているので変数 x3 のみ 1 となっています.

```
RRR-3.BAS x1:  0
RRR-3.BAS x2:  0
RRR-3.BAS x3:  1
                                        RRR-3.BAS
FOR i = 1 TO 10
    x = RND(5)
    IF x <= .3 THEN
        x1 = x1 + 1
    ELSE
        IF x <= .6 THEN
           x2 = x2 + 1
```

```
            ELSE
                x3 = x3 + 1
            END IF
        END IF
    NEXT i
    PRINT x1
    PRINT x2
    PRINT x3
```

③ 続いて [f･7] をおします．再び x2=x2+1 の実行にきてプログラムが止まります．２つ目の乱数は 0.3 をこえ 0.6 以下のため x2 も 1 となっています．

```
 RRR-3.BAS x1:  0
 RRR-3.BAS x2:  1
 RRR-3.BAS x3:  1
                                          RRR-3.BAS
FOR i = 1 TO 10
    x = RND(5)
    IF x <= .3 THEN
        x1 = x1 + 1
    ELSE
        IF x <= .6 THEN
            x2 = x2 + 1
        ELSE
            x3 = x3 + 1
        END IF
    END IF
NEXT i
PRINT x1
PRINT x2
PRINT x3
```

④ 再び [f･7] をおします．x2=x2+1 の実行にきて止まります．３回目は乱数は 0.3 をこえ 0.6 以下のため x2 は 2，４回目は 0.3 以下のため x1 は 1 となっています．５回目の値が 0.3 をこえ 0.6 以下のため x2=x2+1 のステートメントで止まっています．

```
 RRR-3.BAS x1:  1
 RRR-3.BAS x2:  2
 RRR-3.BAS x3:  1
                                          RRR-3.BAS
FOR i = 1 TO 10
    x = RND(5)
    IF x <= .3 THEN
        x1 = x1 + 1
    ELSE
        IF x <= .6 THEN
            x2 = x2 + 1
        ELSE
            x3 = x3 + 1
        END IF
    END IF
NEXT i
PRINT x1
PRINT x2
PRINT x3
```

⑤ 再度 [f･7] をおします．x1 は 2，x2 は 3，x3 は 5 となって終了します．すなわち，３回目に x2=x2+1 が実行されると再びこのステートメントを通ることはありませんので，プログラムの終了まで進みます．

```
RRR-3.BAS x1:  2
RRR-3.BAS x2:  3
RRR-3.BAS x3:  5
```

RRR-3.BAS

```
FOR i = 1 TO 10
    x = RND(5)
    IF x <= .3 THEN
        x1 = x1 + 1
    ELSE
        IF x <= .6 THEN
           x2 = x2 + 1
        ELSE
           x3 = x3 + 1
        END IF
    END IF
NEXT i
PRINT x1
PRINT x2
PRINT x3
```

【例題 65】　関　数

次のプログラムを実行し，STOP のつど，呼び出し関係を確認せよ．プログラムは数 x%を入力し，サブルーチン a1 を呼んで 1 から x%までの和を求め，メインルーチンにもどり，次にファンクション a2 を呼んで 1 から x%までの階乗を求めるものである．プログラム名は RRR-6.BAS とする．

```
DECLARE SUB a1 (z AS INTEGER, zz AS INTEGER)
DECLARE FUNCTION a2 (z AS INTEGER)
INPUT x%
STOP
CALL a1(x%, xx%)
PRINT xx%
PRINT a2(x%)
END


SUB a1 (z%, zz%)
zz% = 0
FOR i = 1 TO z%
    zz% = zz% + i
NEXT i
STOP
END SUB

FUNCTION a2 (z%)
y = 1
FOR i = z% TO 1 STEP -1
    y = y * i
NEXT i
a2 = y
STOP
END FUNCTION
```

```
? 5
 15
 120
```

5 を入力の例

【解　説】　◎　関数メニューで関数の呼び出し関係をチェックします．

① f･5 でプログラムを実行します．x%に値を入力した後，4 行目の STOP でプログラムが中断します．次図となり，現在 RRR-6.BAS のプログラムを実行していることがわかります．

```
C/関数

M/RRR-6.BAS
```

② 続いて f・5 をおし，プログラムを続行します．サブルーチン a1 の STOP で中断します．ここで関数メニューを選びます．下図となります．

RRR-6.BAS から呼び出された a1 を実行していることがわかります．

```
C/関数

a1
M/RRR-6.BAS
```

③ f・5 でプログラムを続行します．次はファンクション a2 の STOP で中断します．ここで関数メニューを選びます．下図となり，RRR-6.BAS から呼び出された a2 を実行中であることがわかります．

```
C/関数

a2
M/RRR-6.BAS
```

【例題 66】　関　数

下のプログラムを実行し，サブルーチンの呼び出し関係を確認せよ．ただし，このプログラムはメインルーチンで x%を入力し，サブルーチン a1 を呼び出して 1 から x%までの和を求め，次にサブルーチン a2 を呼び出して，x%の階乗を求めるものである．プログラム名は RRR-7.BAS とする．

```
DECLARE SUB a1 (z%, zz%, zzz%)
DECLARE SUB a2 (z AS INTEGER, zz AS INTEGER)
INPUT x%
STOP
CALL a1(x%, xx%, xxx%)
PRINT xx%
PRINT xxx%
END


SUB a1 (z%, zz%, zzz%)
zz% = 0
FOR i = 1 TO z%
    zz% = zz% + i
NEXT i
STOP
CALL a2(z%, yy%)
zzz% = yy%
END SUB

SUB a2 (y%, yy%)
z% = 1
FOR i = y% TO 1 STEP -1
    z% = z% * i
NEXT i
yy% = z%
STOP
END SUB
```

【解　説】　◎　関数メニューでサブルーチンの呼び出し関係を確認します．

① [f・5] でプログラムを実行します．メインルーチンの4行目のSTOPでプログラムが中断します．ここで関数メニューを選びます．下図となり，プログラムRRR-7.BASが実行中であることがわかります．

```
C/関数
```

```
M/RRR-7.BAS
```

② 次に [f・5] をおして，プログラムを続行します．次にサブルーチンa1のSTOPで中断します．ここで関数メニューを選びます．次の画面となります．RRR-7.BASから呼び出されたa1を実行中であることがわかります．

```
C/関数
```

```
a1
M/RRR-7.BAS
```

③ [f・5] でプログラムを続行します．サブルーチンa2の中のSTOPで中断します．ここで関数メニューを選びます．次の画面となります．a1から呼び出されたa2を実行中であることがわかります．

```
C/関数
```

```
a2
a1
M/RRR-7.BAS
```

II……プログラム編

1章　変数の型・四則演算・累乗・整数除算・剰余

%, !, $, &, #, +, −, *, /, ∧, ¥, MOD, DEF INT/SNG/LNG/DBL

【例題1】　変数の型

整数，長整数，単精度浮動小数点数，倍精度浮動小数点数，文字を1つずつ入力して表示するプログラムをつくれ．

【解　説】　◎　変数の型を学びます．

① Quick BASICではC言語やFORTRANのように変数の型を明確にします．

② 変数名に%がつくと整数型で−32,768〜+32,767の値を扱います．

③ 変数名に&がつくと長整数型で−2,147,483,648〜+2,147,483,647の値を扱います．

④ 変数名に!がつくと単精度浮動小数点で−3.402823E+38から−1.175494E−38と+1.175494E−38から+3.402823E+38までの値を扱います．7桁が有効です．

⑤ 変数名に#がつくと倍精度浮動小数点で−1.797693134862316D+308から−2.225073858507201D−308までと+2.225073858507201D−308から−1.797693134862316D+308の値を扱います．15または16桁が有効です．

⑥ 変数名に$がつくと文字型で文字列を扱います．最大32,767文字まで代入できます．

⑦ 変数名が同じでも型宣言文字が異なると別変数として扱われます．たとえば，a%とa#は別の変数として使えます．

⑧ a%=156によって変数a%に156が代入されます．a%=156.3のように異なる型を代入するとa%には%で決められた型すなわちこの場合は整数として156を代入します．

⑨ 数値変数に文字を代入することはできません．また，逆に文字変数に数値を代入できません．ただし，数値を""で囲んで代入すればそれは数値ではなく文字として代入されます．

正しい例	誤
a%=156	a%="ABC"
a%=ABC	a%=ABC……ABCが変数ならその値をa%に代入する ABCが文字なら誤り

a$="156"　　　　a$=156……156 が数値なら a$=156 は誤り
a$="156"とすると 156(いちごろく)の
数字が a$に代入される.

⑩ PRINT 変数名で変数の値が表示されます. ;は改行をせずにそのまま続けて表示されます. 文の最後に;がないときは改行されます.

⑪ LPRINT では印刷されます.

【プログラム例】

```
a% = 156 ──────────────┬ a%, a&, a!, a#, a$に値を代入
a& = 123456789         │
a! = 1213.456          │
a# = 123456.78945#     │
a$ = "Quick Basic"─────┘
PRINT a%; a&; a!; a#; a$ ──── 表示
END
```

【結　果】

```
156  123456789  1213.456  123456.78945 Quick Basic
```

【例題 2】　長整数型変数の和

長整数型変数 a& を 1,234,567 とし, b& を 988,777,666 として和を求めて表示するプログラムをつくれ.

【解　説】　◎　和は+で表わします.

① 和は+です. PRINT 文の中で a&+b& のように表わすことができます. a& と b& の和が表示されます.

② c&=a&+b& のように結果を左辺の変数に代入し, c& を表示することもできます.

③ PRINT 文の中の""の中の文字はそのまま表示されます.

④ ;は改行をせずに続けて表示されます.

【プログラム例】

```
a& = 1234567 ──────────┬ 値の代入
b& = 988777666 ────────┘
PRINT a&; "+"; b&; "="; a& + b& ──── 和を表示
END
```

【結　果】

```
1234567 + 988777666 = 990012233
```

【例題 3】　倍精度浮動小数点数の積

52.32145698 と 1.4585424 の積を求めるプログラムをつくれ.

【解　説】　◎　倍精度浮動小数点数の積を求めます.

① 52.32145698 を a#，1.4585424 を b#に代入します．積は*で示します．下の式で a#と b#の積を c#に代入します．

c#=a#*b#

② 四則演算は+，-，*，/です．

【プログラム例】

```
a# = 52.32145698# ─────────┐値を a#と b#に代入
b# = 1.4585424# ───────────┘
c# = a# * b# ──────────────── a#と b#を c#に代入
PRINT a#; "*"; b#; " = "; c# ─── 結果の表示
END
```

【結　果】

```
52.32145698 * 1.4585424  =  76.31306343510596
```

> **【例題 4】　累　乗**
>
> 3 の 8 乗の値を求めるプログラムをつくれ．

【解　説】　◎　累乗は^で求めます．

① x%は整数型変数です．x%を 3 とします．

② b%=x%^8 では x%の値の 8 乗の値を b%に代入します．^は累乗を示します．この例では 8 乗です．5.3 乗のように浮動小数点数も使えます．たとえば，b!=a!^c!です．

③ b&=x%^8 も使えます．x%または 8 の部分が大きくなると b& の方がオーバーフローしない場合が多くなります．

【プログラム例】

```
x% = 3
b% = x% ^ 8 ──────────── 3⁸ を b%へ代入
PRINT x%; "^ 8="; b% ──── 表示
END
```

【結　果】

```
3 ^ 8= 6561
```

> **【例題 5】　整数除算と剰余　¥，MOD**
>
> 98765589÷5476555 の商と剰余を求めるプログラムをつくれ．

【解　説】　◎　整数除算と剰余を求めます．

① c&=a&¥b& は長整数 a&÷b& の商を c& へ代入します．

② ¥は整数除算です．16¥3 は 5 です．16.7¥3.2 は 17¥3 と同じで 5 です．小数点以下は四捨五入されてから計算されます．

③ d&=a&　MOD b& では a&÷b& の剰余を d& へ代入します．MOD は整数除算の剰余です．

【プログラム例】

```
a& = 98765589 ─────────┐─ a&, b& に値を代入
b& = 5476555 ──────────┘
c& = a& ¥ b& ──────────── a& ¥ b& を求めて c& へ代入
d& = a& MOD b& ────────── a& ¥ b& の剰余を d& へ代入
PRINT a&; "/"; b&; " = "; c&; "ｱﾏﾘ"; d& ─── 結果の表示
END
```

【結　果】

```
98765589 / 5476555  =  18 ｱﾏﾘ 187599
```

【例題 6】　型の異なる変数型へ結果を代入

589 を a%, 246 を b%として a%/b%を求めて c#に代入して，表示するプログラムをつくれ.

【解　説】　◎　整数型どうしの割り算の結果を倍精度浮動小数点型変数に代入します.

① a%を 589, b%を 246 として a%/c%を求めます. このとき計算結果を c%に代入すると c%は 2 です. c#に代入すると倍精度浮動小数点型の値 2.394309043884277 が代入されます. 計算結果は代入される変数の型によって決まります.

		a%=589, b%=246 としたときの値
整数型	c%=a%/b%	2
長整数	c&=a%/b%	2
単精度浮動小数点数	c!=a%/b%	2.394309
倍精度浮動小数点数	c#=a%/b%	2.394309043884277

【プログラム例】

```
a% = 589 ─────────┐─ a%, b%に値を代入
b% = 246 ─────────┘
c# = a% / b% ──────── a%/b%を c#に代入
PRINT a%; "/"; b%; " = "; c# ─── 結果の表示
END
```

【結　果】

```
589 / 246  =  2.394309043884277
```

【例題 7】　変数の型宣言

変数 A を整数型, 変数 B を長整数型, 変数 C を単精度浮動小数点数型, 変数 D を倍精度浮動小数点数型, 変数 E を文字型と宣言して，a～e を下の値として表示するプログラムをつくれ.

A　168, B　2583296, C　15.389, D　123456789

E　アイウエオ

【解　説】　◎　変数の型宣言をします.

① DEFINT A は変数 A（大文字小文字の区別はないから a も同じ）を INT 型, 整数型と宣言しま

す．実際には変数Aのみでなく A で始まるすべての変数を整数型とします．したがって，今後a%とせずaのみで整数型を示します．

② DEFLNG B は同様に変数Bを LNG 型，長整数型とします．

③ DEFSNG C は同様に変数Cを SNG 型，単精度浮動小数点数型とします．

④ DEFDBL D は同様に変数Dを DBL 型，倍精度浮動小数点数型とします．

⑤ DEFSTR E は同様に変数Eを STR 型，文字型とします．

【プログラム例】

```
DEFINT A ──────────── A～Eの型宣言
DEFLNG B
DEFSNG C
DEFDBL D
DEFSTR E
a = 168 ───────────── a～eに値の代入
b = 2583296
c = 15.389
d = 123456.789#
e = "ｱｲｳｴｵ"
PRINT a; b; c; d; e ── 表示
END
```

【結　果】

```
168   2583296   15.389   123456.789 ｱｲｳｴｵ
```

【例題8】　文字型を範囲で指定と文字変数の和

A，B，C で始まる変数を文字型と宣言し，alanguage を"Microsoft"，blanguage を"□Quick c"，clanguage を"□Quick BASIC"として alanguage+blanguage，alanguage+clanguage を表示するプログラムをつくれ．

【解　説】　◎　A から C までの文字で始まる変数名を文字型とします．また，文字変数の和を求めます．

① DEFSTR A-C は A から C まですなわち，A，B，C で始まる変数名を文字型と定義します．

② したがって，alanguage，blanguage，clanguage は各々a，b，c で始まっていますから文字型変数です．

③ alanguage+blanguage は文字変数の和で各々の文字列をつなぎます．

【プログラム例】

```
DEFSTR A-C ──────────────────── A～Cを文字型で宣言
alanguage = "Microsoft" ─────── 文字列を代入
blanguage = " Quick C"
clanguage = " Quick Basic"
PRINT alanguage + blanguage ─── 表示．和は文字列をつなぐ
PRINT alanguage; clanguage
END
```

【結　果】

```
Microsoft Quick C
Microsoft Quick Basic
```

【例題 9】　配列の型宣言

配列 a(10)を文字列配列として，a(0)〜a(9)に下のデータを代入して，次に"T"で始まる文字列を表示するプログラムをつくれ.

データ：Rome, Washington, Tokyo, London, Hongkong,
　　　　Chicago, Taipei, Paris, Bon, Moscow

【解　説】　◎　配列の型宣言をします.

① 文字型配列は次のように宣言します. 配列 a(0)〜a(9)が文字型であると宣言するとともに，各文字列の長さは 10 文字となります. 配列についてはⅡ編 8 章で扱います.

DIM	a(10)	AS STRING	*	10
配列宣言	配列名 a$(10)と $をつける 必要はない	文字列として 宣言		10 文字の 文字列として固定

② a(0)〜a(9)に文字列を代入します. FOR 文によるくり返しはⅡ編 5 章で扱います. a(0)は Rome, a(1)は Washington ですが，各文字列は 10 文字ですから a(0)は"Rome□□□□□□"となります（□は空白）. a(1)は"Washington"です.

③ ASC（文字）は文字のアスキーコードを示す値です. たとえば，"A"，"America"は先頭文字"A"のアスキーコード 65 です.

【プログラム例】

```
DIM a(10) AS STRING * 10 ──────── 各10文字の文字列の文字列配列 a(9)を宣言
FOR i = 0 TO 9 ───────────┐ a(0)〜a(9)にデータを読み込む
    READ a(i)              │
NEXT i ───────────────────┘
FOR i = 0 TO 9 ───────────┐ "T"で始まる文字列を表示
    IF ASC(a(i)) = 84 THEN PRINT a(i)
NEXT i ───────────────────┘
END
DATA Rome,Washington,Tokyo,London,Hongkong
DATA Chicago,Taipei,Paris,Bon,Moscow
```

【結　果】

```
Tokyo
Taipei
```

【例題 10】　配列を整数型で宣言

配列 a(10)を整数型で宣言し，下のデータを読み次に，その値をアスキーコードとする文字を順に表示するプログラムをつくれ.

データ：87, 65, 83, 72, 73, 78, 71, 84, 79, 78

【解　説】　◎　配列を整数型で宣言します.

① 配列 a(10)を整数型で宣言するときは次のようにします．また，その他の型の場合も INTEGER の部分を変えるだけです．文字型については【例題 9】で扱っています．配列についてはII編 8 章で扱います.

```
DIM  a(10)   AS┬INTEGER……整数型で宣言
               ├LONG……………長整数型で宣言
               ├SINGLE………単精度浮動小数点数型で宣言
               └DOUBLE………倍精度浮動小数点数型で宣言
```

② CHR$(値)はその値をアスキーコードとする文字を持ちます．たとえば，CHR$(65)は A です.

【プログラム例】

```
DIM a(10) AS INTEGER ──────────────配列 a(9)を整数型で宣言
FOR i = 0 TO 9 ────────────────┬─データの読み込み
   READ a(i)                   │
NEXT i ────────────────────────┘
FOR i = 0 TO 9 ────────────────┬─表示
   PRINT CHR$(a(i));           │
NEXT i ────────────────────────┘
END
DATA 87,65,83,72,73,78,71,84,79,78
```

【結　果】

```
WASHINGTON
```

〔演習問題〕

(1)　長整数 x& を 123456789，y& を 9874563 として x&-y& を b& に代入し，結果を表示するプログラムをつくれ.

(2)　a& を 85236987，b& を 654865 として a&/b& を c& に求め結果を表示するプログラムをつくれ.

(3)　(2) と同じ式の値を c# に求めて表示するプログラムをつくれ.

(4)　a%を 1234，a!を 123.456，a& を 123456789，a# を 123.45678912，a$を"Quick Basic"として，各々を表示するプログラムをつくれ.

(5)　A と U で始まる変数を倍精度浮動小数点数型，R から S までで始まる変数名を単精度浮動小数点数型と宣言して，uriage1 を 1986500，uriage2 を 856200，uriage3 を 752500,

arari1を350200, arari2を97600, arari3を102500として, ritsu1にuriage1に対するarari1の率, seihin1にuriage2に対するarari2の率, seihin2にuriage3に対するarari3の率を小数点第1位まで切りすてで求めるプログラムをつくれ.

(6)　25638.9÷1965.3の整数除算をするプログラムをつくれ.

(7)　3892÷19の商と剰余を求めるプログラムをつくれ.

(8)　3^9を求めるプログラムをつくれ.

(9)　整数x!を3.5, y!を2.5としてx!$^{y!}$を求めるプログラムをつくれ.

(10)　186.3292654#*5%を求め結果を単精度実数で表示するプログラムをつくれ.

(11)　配列a(10)を単精度実数型で宣言し, 次の数値データを10個読み込んで表示するプログラムをつくれ.

データ;87, 65, 83, 72, 73, 78, 71, 84, 79, 78

(12)　A, B, Cで始まる変数名を文字型と宣言し, America, Austria, Britain, Burma, Canada, Chinaに各々次の文字列を代入して表示するプログラムをつくれ.

USA, Europe, England, Asia, North America,
Mainland China

(13)　x, y, zで始まる変数を整数型変数と宣言し, a!=38.29, b!=11.33, c!=8.76としてa!*b!の値をxに, a!÷c!の値をyにb!*c!の値をzに代入して表示するプログラムをつくれ.

〔解　答〕

(1)　**【プログラム例】**

```
x& = 123456789 ──────────┐── 値の代入
y& = 9874563 ────────────┘
b& = x& - y& ──────────────── 差
PRINT x&; "-"; y&; "="; b& ── 結果の表示
END
```

【結　果】

```
123456789 - 9874563 = 113582226
```

(2)　**【プログラム例】**

```
a& = 85236987 ───────────┐── a&とb&に値を代入
b& = 654865 ─────────────┘
c& = a& / b& ──────────────── a&/b&をc&に代入
PRINT a&; "/"; b&; "="; c& ── 結果の表示
END
```

【結　果】

```
85236987 / 654865  =  130
```

(3)　【プログラム例】

```
a& = 85236987 ─────────┐─a& と b& に値を代入
b& = 654865 ───────────┘
c# = a& / b& ─────────── a&/b& を c# に代入
PRINT a&; "/"; b&; "="; c# ── 結果の表示
END
```

【結　果】

```
85236987 / 654865  =  130.1596313744054
```

(4)　【プログラム例】

```
a% = 1234 ──────────── 整数型変数に値を代入
a! = 123.456 ───────── 単精度型実数に値を代入
a& = 1213456789 ────── 長整数型変数に値を代入
a# = 123.45678912# ─── 倍精度型実数に値を代入
a$ = "Quick Basic"──── 文字型変数に値を代入
PRINT a%; a!───────┐── 表示
PRINT a&; a#       │
PRINT a$ ──────────┘
END
```

【結　果】

```
 1234  123.456
 1213456789  123.45678912
Quick Basic
```

(5)　【プログラム例】

```
DEFDBL A, U ─────────────────────────────── A, U を倍精度浮動小数点数型として宣言
DEFSNG R-S ──────────────────────────────── R, S を単精度実数として宣言
uriage1 = 1986500 ─────────────────────┐─── 初期値の代入
uriage2 = 856200                       │
uriage3 = 752500                       │
arari1 = 350200                        │
arari2 = 97600                         │
arari3 = 102500 ───────────────────────┘
ritsu1 = INT(arari1 / uriage1 * 1000) / 10 ─┐─ 率の計算
seihin1 = INT(arari2 / uriage2 * 1000) / 10 │
seihin2 = INT(arari3 / uriage3 * 1000) / 10─┘
PRINT ritsu1; "%   "; seihin1; "%   "; seihin2; "%   "──── 結果の表示
END
```

【結　果】

```
17.6 %    11.3 %    13.6 %
```

(6)　【プログラム例】

```
PRINT 25638.9 ¥ 1965.3 ──── 整数除算
END
```

【結　果】

```
13
```

(7)　【プログラム例】

```
a% = 3892
b% = 19
x1% = a% ¥ b% ――― 商
x2% = a% MOD b% ―― 剰余
PRINT x1%, x2%
END
```

【結　果】

```
204             16
```

(8)　【プログラム例】

```
a% = 3
b% = a% ^ 9 ――― 3⁹ を b%に代入
PRINT b%
END
```

【結　果】

```
19683
```

(9)　【プログラム例】

```
x! = 3.5 ――――┐
              ├ 値の代入
y! = 2.5 ――――┘
PRINT x! ^ y! ――― 結果の表示
END
```

【結　果】

```
22.91765
```

(10)　【プログラム例】

```
x# = 186.3292654#
x% = 5
x! = x# * x% ――― x#*x%を x!に代入
PRINT x!
END
```

【結　果】

```
931.6463
```

(11)　【プログラム例】

```
DIM a(10) AS SINGLE ――――――――― 配列の宣言
FOR i = 0 TO 9 ――――――――――┬ データの読み込み
    READ a(i)                  │
NEXT i ――――――――――――――┘
FOR i = 0 TO 9
    PRINT a(i); ―――――――――┬ 表示
NEXT i ――――――――――――――┘
END
DATA 87,65,83,72,73,78,71,84,79,78
```

【結　果】

```
87  65  83  72  73  78  71  84  79  78
```

(12) 【プログラム例】

```
DEFSTR A-C ──────────────────型宣言
america = "USA"────────────┬─値の代入
austria = "Europe"         │
britain = "England"        │
burma = "Asia"             │
canada = "North America"   │
china = "Mainland China"───┘
PRINT america, austria ────┬─表示
PRINT britain, burma       │
PRINT canada, china ───────┘
END
```

【結　果】

```
USA             Europe
England         Asia
North America   Mainland China
```

(13) 【プログラム例】

```
DEFINT X-Z ──────────x, y, z で始まる変数名を整数型に宣言
a! = 38.29 ─────────┬─値の代入
b! = 11.33          │
c! = 8.76 ──────────┘
x = a! * b! ────────┬─積, 商, 積を x, y, z に代入
y = a! / c!         │
z = b! * c! ────────┘
PRINT x, y, z
END
```

【結　果】

```
434             4               99
```

2章 定　　数

CONST

【例題 11】　記号定数

Pi を定数 3.14159 として，半径 30 の円の面積と円周長を求めるプログラムをつくれ．

【解　説】 ◎ 記号定数を使います．

① 定数に記号をつけ，常にその値を保ちます．

② CONST Pi!=3.14159 とすると Pi の値は常に 3.14159 に固定されます．途中で Pi=5.2 のように値を入れかえることはできません．つまりうっかり値を変えてしまうといったミスはおこりません．変数を使うとそういうことがおこることがあります．また，3.14159 のように値の桁が多い場合は記号定数に代入しておいた方が使いやすくなります．

③ CONST Pi!=3.14159 の”!”によって Pi の型を単精度実数型とします．

④ CONST Pi!=3.14159 では Pi が定数の名前で，！は定数のデータ型を示す型宣言文字です．

⑤ 記号定数で使われた名前，本例では Pi は以降，変数として使用できません．Pi%，Pi!，Pi#，Pi$のいずれも使えません．

【プログラム例】

```
CONST pi! = 3.14159 ———定数 3.14159 を pi!に固定する
s! = 30 * 30 * pi ———面積の計算
l! = 2 * 30 * pi———円周長の計算
PRINT s! ————┐結果の表示
PRINT l! ————┘
END
```

【結　果】

```
2827.431
188.4954
```

【例題 12】　整数型の記号定数

九九の表をつくるプログラムをつくれ．

【解　説】 ◎ 整数型の記号定数を使います．

① CONST ic%=9 では ic%を 9 に固定します．%は整数型を示します．

② FOR i%=1　TO ic%

```
FOR j%=1  TO ic%
  ～
```

によってi%を1から9，j%を1から9までループします．くり返しはII編5章で扱います．

③　五五の表にしたいというときはCONST ic%=5となおすことでi%を1から5，j%を1から5までのループとします．このように記号定数を使うことで訂正が簡単にいくことがあります．特にic%をプログラムの中でたくさん使っている場合それをさがすのは大変です．

【プログラム例】

```
CONST ic% = 9 ―――――――――――――――――――― ic%を9に固定する
FOR i% = 1 TO ic% ――――――――――――――――― iを1から9までループ
    FOR j% = 1 TO ic%――――――――――――――― jを1から9までループ
        PRINT USING "## "; i% * j%; ――┬ i*jを表示
    NEXT j% ――――――――――――――――――――――┘
PRINT ――――――――――――――――――――――――――――┬ 改行
NEXT i% ――――――――――――――――――――――――――┘
END
```

【結　果】

```
 1  2  3  4  5  6  7  8  9
 2  4  6  8 10 12 14 16 18
 3  6  9 12 15 18 21 24 27
 4  8 12 16 20 24 28 32 36
 5 10 15 20 25 30 35 40 45
 6 12 18 24 30 36 42 48 54
 7 14 21 28 35 42 49 56 63
 8 16 24 32 40 48 56 64 72
 9 18 27 36 45 54 63 72 81
```

【例題13】　文字型記号定数

m$を文字定数"Microsoft"で固定し，m$と"□Quick C"，m$と"□Quick BASIC"を表示するプログラムをつくれ．

【解　説】　◎　文字型の記号定数を使います．

①　次の形で変数m$を"Microsoft"に固定します．

```
CONST m$="Microsoft"
```

【プログラム例】

```
CONST m$ = "Microsoft"―――――――――― 文字定数"Microsoft"をm$に固定する
m1$ = m$ + " Quick C"――――――――――┬ m$と文字列の和をm1$とm2$に代入
m2$ = m$ + " Quick Basic"――――――┘
PRINT m1$ ――――――――――――――――――――┬ 結果の表示
PRINT m2$ ――――――――――――――――――――┘
END
```

【結　果】

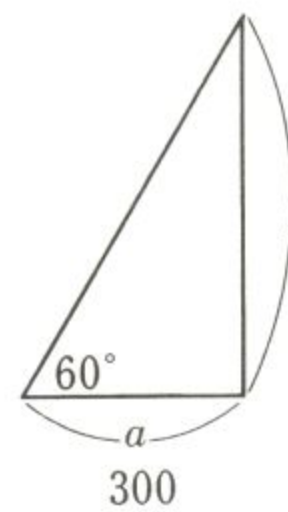
```
Microsoft Quick C
Microsoft Quick Basic
```

〔演習問題〕

(14)　x%を定数15として，次の15個のデータを読んで，その和を求めるプログラムをつくれ.

データ：　　1.25,2.56,3.25,4.55,3.21
　　　　　　4.55,2.36,3.54,1.56,7.51
　　　　　　8.21,9.56,8.12,9.21,2.09

(15)　x!を1.7321として次図のlを求めるプログラムをつくれ．ただし，1.7321は次図のaを1としたときのlの長さである.

60°　l　a　300

(16)　a$を"□C"として，Turbo C, Microsoft C, Quick C, Run C, Lattice Cと表示するプログラムをつくれ.

〔解　答〕

(14)【プログラム例】

```
CONST x% = 15 ──────────── x%を15定数とする
FOR i% = 1 TO x% ────────┐ データの読み込みと和
    READ z               │
    zz = zz + z          │
NEXT i% ─────────────────┘
PRINT zz ────────────────── 和を表示
END
DATA 1.25,2.56,3.25,4.55,3.21
DATA 4.55,2.36,3.54,1.56,7.51
DATA 8.21,9.56,8.12,9.21,2.09
```

【結　果】

71.53

(15)【プログラム例】

```
CONST x! = 1.7321──── x!を定数1.7321とする
l = 300 * x!───────── lを求める
PRINT l ───────────── lの表示
END
```

【結　果】

519.63

(16)【プログラム例】

```
CONST a$ = " C"─────────────a$を定数"□C"とする
FOR i = 1 TO 5 ─────────────5回のくり返し
    READ aa$ ───────────────文字列データを読む
    PRINT aa$ + a$ ─────────読み込んだ文字列と"□C"を加えて表示
NEXT i
END
DATA Turbo,Microsoft,Quick,Run,Lattice
```

【結　果】

```
Turbo C
Microsoft C
Quick C
Run C
Lattice C
```

3章　入力・データの読み込み

INPUT, INPUT$, INPUT¥, LINEINPUT, INKEY$,
READ, DATA, RESTORE

【例題14】　INPUTによる入力

整数を２つ入力し，次に実数を２つ，次に文字列を２つ入力して表示するプログラムをつくれ.

【解　説】　◎　数値と文字列を入力します.

① INPUT変数名で変数名に値を代入します. 入力するデータの型にあわせた変数名を使います.

〔例〕

INPUT	x%	整数を１つx%に入力
INPUT	x!	単精度実数を１つx!に入力
INPUT	x$	文字列を１つx$に入力

② プログラムが実行されると?が表示されますのでデータを入力します. データの型が異なると指定された型で入力されます. ただし，数値型変数へ文字列の入力はできませんので"再入力して下さい"の指示がでます.

③ ２つ以上のデータを入力するときは変数名をカンマで区切ります.

〔例〕

INPUT	x%, y%	整数を２つ入力
INPUT	x!, y!, z!	単精度実数を３つ入力
INPUT	x$, y#, z$	文字列１つ，倍精度実数を１つ，文字列を１つ入力

④ プログラムを実行すると?が表示されますので，次のようにデータを区切って，変数の型と順番および数をそろえて入力します.

データ，データ，データ ⏎ またはデータ□データ□データ ⏎ （□はスペース）

⑤ 型が異なったり，数が違っていたりすると再入力を促します.

【プログラム例】

```
INPUT x%, y% ———整数２つ入力
INPUT x, y  ———実数２つ入力
INPUT x$, y$———文字２つ入力
PRINT x%, y% ———整数２つ表示
PRINT x, y  ———実数２つ表示
PRINT x$, y$———文字２つ表示
END
```

【結　果】

```
? 25,34                          ――― 25, 34↵
? 1.26,35.287                    ――― 1.26, 35.287↵
? ABC,ZYX                            ABC, ZYX↵ の例
 25             34
 1.26           35.287
ABC            ZYX
```

【例題 15】　プロンプト文付きの入力

”スウチ(1 カラ 100)　ノ　ニュウリョク”と表示した後に数値を入力して，表示するプログラムをつくれ.

【解　説】　◎　プロンプト文付きで入力

① INPUT 変数名での入力は?が表示されるだけです．次のようにかくと”　”の中の文章を表示しますので入力する内容がわかりやすくなります.

INPUT　　”文字列”;変数名

② PRINT　　”文字列”;

INPUT　　変数名

でも同じ効果がでます.

【プログラム例】

```
INPUT "1 カラ 100 ノ ニュウリョク"; x% ――― 整数の入力
PRINT x% ――――――――――――――――――――― 表示
END
```

【結　果】

```
1 カラ 100 ノ ニュウリョク? 55          55 を入力の例
 55
```

【例題 16】　LINE INPUT による入力

文章を入力し，そのまま表示するプログラムをつくれ.

【解　説】　◎　文章の入力をします.

① LINE INPUT x$はキーからおされた文字・文章をそのまま x$に受けつけます．”や ,もそのまま受けつけます．INPUT x$では ,や”は受けつけません.

【プログラム例】

```
LINE INPUT x$ ――― 入力
PRINT x$ ――――――― 表示
END
```

【結　果】

```
"Quick Basic,Quick C,Quick Fortran" ↵ ――― 入力例
"Quick Basic,Quick C,Quick Fortran"  ――― 表示例　”や,が受けつけられている
```

【例題17】 INPUT$, INPUT¥による入力

キーから8文字と6文字の2文字列を入力して表示するプログラムをつくれ.

【解 説】 ◎ INPUT$とINPUT¥による指定文字数の入力をします.

① 次の形で指定した文字数の文字列を受けつけ，変数に代入します．漢字は2で1文字です.

x$=INPUT$(8)
入力して x$に代入 ── 8バイト，半角文字は8文字 漢字の場合は4文字

y$=INPUT¥(6)
── 6文字，半角でも漢字でも6文字

② エコーバックはありません．おされたキーが指定した文字数になると変数に代入されます．⏎はしません.

【プログラム例】

```
x$ = INPUT$(8)———8文字の入力
y$ = INPUT¥(6)———6文字の入力
PRINT x$, y$
END
```

【結 果】

```
abcdefgh      ijklmn        abcdefghijklmnを入力 前の8文字がx$，後の6文字がy$に代入
あいうえ      おかきくけこ  あいうえおかきくけこを入力 前の4文字がx$，後の6文字がy$に代入
```

【例題18】 キーから1文字入力

キーから文字がおされて入力されるのを待ち，入力されたらその文字・記号を表示するプログラムをつくれ．ただし，"e"がおされたら終了するものとする.

【解 説】 ◎ キー入力のくり返しをします.

① 次の形で1文字を入力し，表示します．"e"が入力されると終了します．DO文についてはⅡ編5章で扱います.

```
DO
x$=INKEY$ ———————— キーより1文字入力
PRINT x$ ————————— 表示
LOOP WHILE x$<>"e" ——— "e"が入力されるとx$<>"e"が偽となりDO LOOPを終了する
```

【プログラム例】

```
DO
x$ = INKEY$———————キー入力
PRINT x$; ————————表示
LOOP WHILE x$ <> "e"——— "e"のとき終了
END
```

【結　果】

```
54jkihkguycfe          5，4，…，e を入力の例
```

【例題 19】　特殊キーの判定

キーをおし，↵ キーのときプログラムを終了し，その他のキーのときはその文字，記号を表示するプログラムをつくれ．

【解　説】　◎　おされたキーを判別します．

① キーがおされると x$=INKEY$で x$にキーのアスキーコードが入ります．したがって，x$のアスキーコードをみると何がおされたかがわかります．

② ↵ キーのアスキーコードは & HOD です．

③ 1F 文，GOTO 文についてはII編6章で扱います．

【プログラム例】

```
10 : x$ = INKEY$ ──────────┐ キー入力
IF x$ = "" THEN GOTO 10 ───┘
PRINT x$; ──────────────────── x$の表示
IF ASC(x$) <> &HD THEN GOTO 10    ↵ のとき終了
END
```

【結　果】

```
Acdeg153vgt fttrh  gf      A c d … □ f t … □□ g f ↵ とキーをおした例
```

【例題 20】　READ によるデータの読み込み

次のデータを配列 r%(15)に読み込んで和を求め表示するプログラムをつくれ．

データ；8，7，2，9，6，5，4，8，2，0，5，2，7，6，7

【解　説】　◎　READ によってデータを読み込みます．

① 次の形でデータを配列 r%(0)～r%(14)に読み込みます．配列についてはII編8章，くり返しについてはII編5章で扱います．

```
FOR i=0  TO  14
  READ r%(i)                    r%(0)r%(1)r%(2)……r%(14)
NEXT i                            |     |     |       |
  :                               |     |     |       |
  :                               |     |     |       |
Data   8, 7, 2, 9, 6, 5, 4, 8     8     7     2       7
Data   2, 0, 5, 2, 7, 6, 7
```

② データ文 Data…はプログラムのどこにかいてもかまいません．前から順に変数にわりあてられます．

③ データ文はいくつかに分割してかくことができます．

④　データ数が変数名より少ないとエラーとなりますが，多いときは余った分は無視されます.

【プログラム例】

```
DIM r%(15) ─────────────── 配列の宣言
FOR i = 0 TO 14 ────────┐─ データの読み込み
    READ r%(i) ──────────┘
NEXT i
FOR i = 0 TO 14 ────────┐─ 和を求める
    t% = t% + r%(i) ─────┘
NEXT i
PRINT t%
END
DATA 8,7,2,9,6,5,4,8 ──────── 結果の表示
DATA 2,0,5,2,7,6,7
```

【結　果】

```
78
```

【例題 21】　READ によるデータの読み込み

下のデータを配列 m$(10)に読み込み"H"で始まるものを表示するプログラムをつくれ.

データ；Tokyo, Hakodate, Morioka, Hirosaki, Hagi

Kochi, Uwajima, Kanazawa, Narita, Kisarazu

【解　説】　◎　文字列データを読み込みます.

①　変数名に文字型変数を使う他は【例題 20】とまったく同じ方法で読み込みます.

②　ASC(m$(i))=72 すなわち"H"で始まる文字列を表示します．if 文についてはⅡ編 6 章で学びます.

【プログラム例】

```
DIM m%(10) ──────────────────────── 配列の宣言
FOR i = 1 TO 10                  │  データの読み込み
    READ m$(i)                   │
NEXT i ──────────────────────────┘
FOR i = 1 TO 10 ─────────────────┐─ "H"で始まる文字列の表示
    IF ASC(m$(i)) = 72 THEN PRINT m$(i)
NEXT i ──────────────────────────┘
END
DATA Tokyo,Hakodate,Morioka,Hirosaki,Hagi
DATA Kochi,Uwajima,Kanazawa,Narita,Kisarazu
```

【結　果】

```
Hakodate
Hirosaki
Hagi
```

【例題 22】　データの再利用

下のデータを使って，a$に Tokyo，b$に Nagoya，c$に Osaka，d$に Nagoya，e$に Osaka，f$に Osaka と読み込むプログラムをつくれ.

データ;Tokyo，Nagoya，Osaka

【解　説】　◎　データを何回か読み込みます.

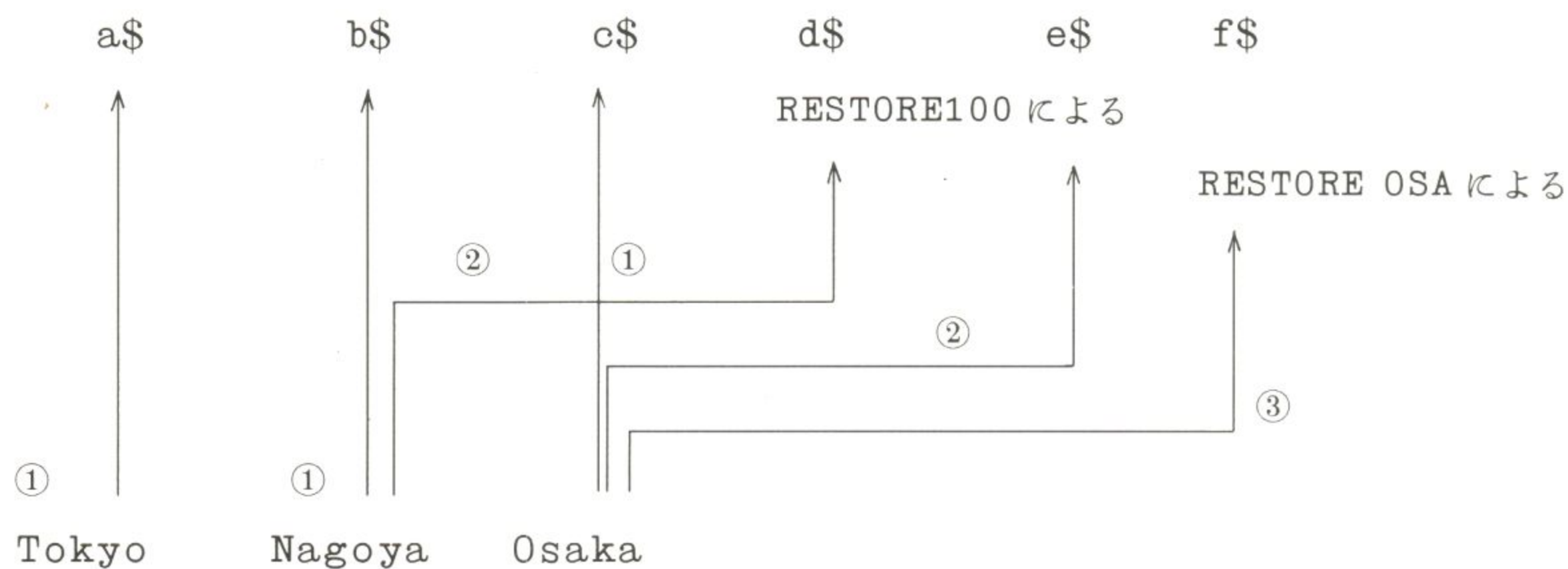
① READ 文では前から順にデータを読んで変数にわりあてますが，RESTORE を使って，順序をかえることができます.

② RESTORE 行番号または RESTORE ラベルによって，その行番号またはラベルへもどって，そこから順にデータを読みます.

a$　b$　c$　d$　e$　f$

RESTORE100 による

RESTORE OSA による

①　①　②　①　②　③

Tokyo　Nagoya　Osaka

①〜③は 1 番目の READ 文〜3 番目の READ 文を示す.

【プログラム例】

```
READ a$, b$, c$ ——————————————————— a$〜c$にデータを読む
RESTORE 100 ———————————————————— データポインタを 100 へもどす
READ d$, e$ ———————————————————— d$, e$にデータを読む
RESTORE osa ———————————————————— データポインタを osa へもどす
READ f$ ————————————————————————— f$にデータを読む
PRINT a$; " "; b$; " "; c$; " "; d$; " "; e$; " "; f$ —— 表示
END
DATA Tokyo
100 DATA Nagoya
osa: DATA Osaka
```

【結　果】

```
Tokyo Nagoya Osaka Nagoya Osaka Osaka
```

【例題 23】　データ文に終わりのマーク

次のデータを読んで配列 x%(10)に格納し，和と平均を求めるプログラムをつくれ．ただし，データ文の最後に 0 をおき，データの終了を示すものとし，この 0 はデータには入れないものとする.

データ;7，6，3，5，9，0

【解　説】　◎　データ文の終わりに 0 をつけてデータの終了を示します.

①　データ数がわかっているときは FOR～NEXT 文のくり返しなどを使って読むことができますが, データ数が不明のときはそれができません. データがないとき READ するとエラーとなります. そこで, データ文の最後に 0 などをつけて, データの終わりを示すと処理がしやすくなります.

②　次はその 1 例です. この形では x(i)に読み込んだデータが 0 のとき読み込みを終了します. この時のデータ数は i-1 です.

```
FOR   i=1  TO  100
        READ x(i)
        1F x(i)=0  THEN EXIT FOR
NEXT i
```

③　次の形でもデータ文の最後の 0 まで読み込みます.

```
DO While x
    c=c+1
    READ x
    x(c)=x
WEND
```

④　データの終わりを示すマークは 0 でなくても, -99999 などデータとして起こりえないものを使います.

【プログラム例】

```
FOR i = 1 TO 10 ─────────────┬─10回のくり返し
    READ y% ─────────────────┼─データを読み込む
    IF y% = 0 THEN EXIT FOR──┼─データが 0 のときループの外へ分岐
              x%(i) = y% ────┼─配列へデータを代入
              t = t + y% ────┼─和を求める
NEXT i ──────────────────────┘
h = t / (i - 1)───────────────平均を求める
PRINT t, h ───────────────────和と平均の表示
END
DATA 7,6,3,5,9,0
```

【結　果】

```
30              6
```

〔演習問題〕

(17)　整数を 1 つ, 文字列を 2 つ入力して順に表示するプログラムをつくれ.

(18)　プロンプト文で"ナマエ　オ　ニュウリョクシテクダサイ"と表示した後, na$に名前を入力して表示するプログラムをつくれ.

(19)　キーをおして, [E] のとき"end"と表示するプログラムをつくれ.

(20)　キーをおして, [ESC] キーのとき escape, [TAB] キーのとき tab と表示するプログラム

をつくれ．ただし，他のキーのときは何も表示しないで終了するものとする．

(21)　キーから10文字を入力して表示するプログラムをつくれ．

(22)　次の文章をキー入力してから表示するプログラムをつくれ．ただし，"も入力するものとする．
"You visited Boston last year, didn't you?"

(23)　下のデータを読んで配列x%(3,7)に代入し，次に横の項の和を求めて，全体を表示するプログラムをつくれ．

3	2	8	6	3	4	2
8	6	7	5	2	8	7
9	4	1	7	2	6	5

(24)　下のデータを使って，a1$にWashington，a2$にChicago，a3$にSalt Lake City，a4$にLos Angeles，a5$にChicago，a6$にSalt Lake City，a7$にWashington，a8$にChicago，a9$にSalt Lake Cityを代入して，順に表示するプログラムをつくれ．

データ;Washington, Chicago, Salt Lake City, Los Angeles

(25)　次のデータを配列x%(15)に読み，和を求めるプログラムをつくれ．ただし，0のときデータは終了するものとする．

データ;9, 7, 8, 5, 2, 7, 6, 8, 1, 3, 2, 4, 7, 4, 7, 0

(26)　キーがおされたら，プログラムを実行してから何秒経過したかを示すプログラムをつくれ．

〔解　答〕

(17) **【プログラム例】**

```
INPUT x% ―――――――――――整数の入力
INPUT x1$, x2$―――――――――文字列を２つ入力
PRINT x%, x1$, x2$――――― 表示
END
```

【結　果】

```
? 555                                              555↵
? Quick C,Quick Basic                              Quick C, Quick BASIC↵ の例
 555            Quick C           Quick Basic
```

(18) **【プログラム例】**

```
INPUT "ﾅﾏｴ ｵ ﾆｭｳﾘｮｸｼﾃｸﾀﾞｻｲ"; na$―――――入力
PRINT na$ ―――――――――――――――――――――表示
END
```

【結　果】

```
ﾅﾏｴ ｵ ﾆｭｳﾘｮｸｼﾃｸﾀﾞｻｲ? 大山太郎        大山太郎↵ の例
大山太郎
```

(19) **【プログラム例】**

```
10 x$ = INKEY$ ─────────────┐─1 文字の入力
   IF x$ = "" THEN GOTO 10 ─┘
IF ASC(x$) = &H45 THEN PRINT "end"──── "E"のとき end と表示
END
```

【結　果】

```
end              "E"をおしたとき
```

(20) **【プログラム例】**

```
10 x$ = INKEY$ ──────────────┐─キーがおされたら x$に代入される,
   IF x$ = "" THEN GOTO 10 ──┘  おされないときはくり返す
IF ASC(x$) = &H1B THEN PRINT "escape"──── ESC キーのときの表示
IF ASC(x$) = &H9 THEN PRINT "tab"──────── TAB キーのときの表示
END
```

【結　果】

```
tab              TAB キーをおした例
escape           ESC キーを入力した例
```

(21) **【プログラム例】**

```
x$ = INPUT$(10)────10 文字の入力
PRINT x$ ──────────表示
END
```

【結　果】

```
Quick C Qu       Q……u の例
```

(22) **【プログラム例】**

```
LINE INPUT x$ ────文章の入力
PRINT x$ ─────────表示
END
```

【結　果】

```
"You visited Boston last year,didn't you ?"────キー入力
"You visited Boston last year,didn't you ?"────表示
```

(23) **【プログラム例】**

```
DIM x%(3, 8) ──────────────────────────┐─データを 2 次元配列へ読み込み横の和を求める
FOR i = 0 TO 2                          │
    FOR j = 0 TO 6                      │
        READ x%(i, j)                   │
        x%(i, 7) = x%(i, 7) + x%(i, j)──┤─横の和を求める
NEXT j, i ──────────────────────────────┘
```

```
FOR i = 0 TO 2 ─────────────┐ 2次元の表を表示
    FOR j = 0 TO 7          │
        PRINT x%(i, j);     │
    NEXT j ─────────────────┘
PRINT
NEXT i
END
DATA 3,2,8,6,3,4,2
DATA 8,6,7,5,2,8,7
DATA 9,4,1,7,2,6,5
```

【結　果】

```
3   2   8   6   3   4   2   28
8   6   7   5   2   8   7   43
9   4   1   7   2   6   5   34
```

(24)　【プログラム例】

```
READ a1$, a2$, a3$, a4$ ──────── a 1$～a 4$にデータを読み込み
RESTORE toshi ────────────────── toshi へデータポインタをもどす
READ a5$, a6$ ────────────────── a 5$, a 6$の読み込み
RESTORE ──────────────────────── 先頭へデータポインタをもどす
READ a7$, a8$, a9$ ───────────── a 7$～a 9$の読み込み
PRINT a1$; " "; a2$; " "; a3$ ─┐ 結果の表示
PRINT a4$; " "; a5$; " "; a6$  │
PRINT a7$; " "; a8$; " "; a9$ ─┘
END
DATA Washington
toshi: DATA Chicago,Salt Lake City,Los Angeles
```

【結　果】

```
Washington Chicago Salt Lake City
Los Angeles Chicago Salt Lake City
Washington Chicago Salt Lake City
```

(25)　【プログラム例】

```
DIM x%(15)
FOR i = 0 TO 14 ──────────────── 15回のくり返し
    READ y% ──────────────────── データの読み込み
    IF y% = 0 THEN EXIT FOR ──── データが0のときループの外へ分岐
    x%(i) = y% ───────────────── 配列へデータを代入
    t = t + y% ───────────────┬─ 和を求める
NEXT i ───────────────────────┘
PRINT t ───────────────────────── 和の表示
END
DATA 9,7,8,5,2,7,6,8
DATA 1,3,2,4,7,4,7,0
```

【結　果】

```
80
```

(26) **【プログラム例】**

```
x1 = TIMER ―――――――――― 現在の秒を代入
DO ――――――――――――――┐ キーがおされるのを待つ
LOOP WHILE INKEY$ = ""――┘
x2 = TIMER ―――――――――― 現在の秒を代入
PRINT x2 - x1 ――――――― 経過時間を表示
END
```

【結　果】

```
15
```

4章　書式指定の表示と印刷

PRINT USING, LPRINT USING

【例題 24】　文字の書式指定表示

a$を"Quick Basic"としてa$を①12桁で，②"ゲンゴ□ハ□"の後に15桁で，③先頭1文字を，④5桁を表示し，次に3スペースの後にa$全体を表示するプログラムをつくれ．

【解　説】　◎　文字列を書式指定で表示します．

① PRINT USINGの後に書式を指定します．

PRINT USING　"&□□□□&";a$
　6桁
　& &でかこむと&&を含む桁数でa$を表示．
　この場合6桁です．文字は左づめです．

PRINT USING　"ゲンゴ　ハ　&　　　&";a$
　文字はそのまま表示
　桁数が足りないときは&&で指示された桁のみ表示

PRINT USING　"!";a$
　先頭1文字を表示

PRINT USING　"@";a$
　桁数に関係なく文字列を表示

② PRINTをLPRINTとすると印刷されます．

【プログラム例】

```
a$ = "Quick Basic"
PRINT USING "&          &"; a$ ──────── 12桁で表示
PRINT USING "ｹﾞﾝｺﾞ ﾊ &             &"; a$ ── 文字の後に表示
PRINT USING "!"; a$ ─────────────── 先頭1文字の表示
PRINT USING "&   &   @"; a$; a$ ───── 5文字と全体を表示
END
```

【結　果】

```
Quick Basic
ｹﾞﾝｺﾞ ﾊ Quick Basic
Q
Quick   Quick Basic
```

【例題 25】　数値の書式指定

x%を 12345，y!を 1234.56，z%を-9876 として①x%と z%を 6 桁で，②y!を全体を 7 桁小数点以下 2 桁と全体を 6 桁小数点以下 1 桁で，③x%を全体 6 桁で先頭と最後に+をつけて，④z%を全体を 7 桁で先頭と最後に-をつけて表示するプログラムをつくれ．

【解　説】　◎　数値の書式指定をします．

① 数値の書式指定も PRINT USING の後に書式をかきます．

② 次のような指定となります．

```
PRINT USING  "######□□□######";x%;z%
```



```
PRINT USING  "####.##□□####.#";y!;y!
```



```
PRINT USING  "+######  ######+";x%;x%
```



```
PRINT USING  "##-";z%
```


【プログラム例】

```
x% = 12345 ───────────────── 値の代入
y! = 1234.56
z% = -9876
PRINT USING "######  ######"; x%; z% ──── 6 桁で表示
PRINT USING "####.##  ####.#"; y!; y! ──── 小数点以下 2 桁と 1 桁で表示
PRINT USING "+######  ######+"; x%; x% ──── +を先頭と最後につけて表示
PRINT USING "#######  #######-"; z%; z% ──── -を先頭と最後につけて表示
END
```

【結　果】

```
 12345    -9876
1234.56  1234.6
 +12345    12345+
  -9876     9876-
```

【例題 26】　＊と¥，カンマ付き表示

x%を 12345 として①全体を 8 桁で，8 桁未満のときは先頭を*をうめて，②全体を 9 桁で，先頭に¥をつけ，小数点以下 2 桁で，③全体を 9 桁で先頭に¥をつけ，9 桁未満のときは*でうめて，④全体を 8 桁で 3 桁のカンマ付きで表示するプログラムをつくれ．ただし，桁数には*，¥，．，，も含めるものとする．

【解　説】　◎　*，¥，カンマをつけます．

① 次の書式で*をつけます．

```
PRINT USING  "**######";x%
```

全体 8 桁，8 桁未満のときは先頭を*でうめる．

② 次の書式で¥をつけます．

```
PRINT USING  "¥¥####.##";x%
```

全体 9 桁，先頭に¥がつく

③ 次の書式で¥をつけます．桁数が少ないときは先頭を＊でうめます．

```
PRINT USING  "**¥######";x%
```

全体 9 桁，先頭に¥がつく
9 桁未満のときは*でうめる．

④ 次の書式で 3 桁ごとの，をつけます．

```
PRINT USING  "####,###";x%
```

全体 8 桁
3 桁ごとのカンマ付き

【プログラム例】

```
x% = 12345
PRINT USING "**######"; x% ───── *でうめて表示
PRINT USING "¥¥####.##"; x%───── ¥つきで表示
PRINT USING "**¥######"; x% ───── ¥つき*でうめて表示
PRINT USING "####,###"; x% ───── 3 桁ごとのカンマ付き表示
END
```

【結　果】

```
***12345
¥12345.00
***¥12345
  12,345
```

【例題 27】　指数書式指定表示

x!を 12.345 として①0.---E--の形の指数で，②1.---E--の形の指数で，③値の前に+符号をつけた指数と E の後を 3 桁とした指数で，④先頭に#，最後に_（アンダスコア）をつけて，⑤全体を####の指定で，ただし，桁不足のときは先頭に%をつけて，各々表示するプログラムをつくれ.

【解　説】　◎　指数書式の指定をします.

① 次の書式で指数の書式を指定します.

```
PRINT USING  "#.###^^^^";x!
```

指数，小数点以上なし
小数点以下 3 桁，指数部は
E+02 のように 4 桁

```
PRINT USING  "#.###^^^^^";x!
```

指数，指数部は
E+001 のように 5 桁

② 次の書式は特殊なものです.

```
PRINT USING  "_###.###__";x!
```

__ → _が 1 つ表示される

_# → _の後の 1 文字はそのまま表示，#が表示される.
この#は桁を示す#ではない

```
PRINT USING  "%####";x!
```

% → 桁数が足りないとき%を表示して示す

【プログラム例】

```
x! = 12.345
PRINT USING "#.###^^^^"; x! ────────── 指数表示，指数部 4 桁
PRINT USING "##.##^^^^"; x!────────── 指数表示，指数部 4 桁
PRINT USING "+#.###^^^^  +#.###^^^^^"; x!; x!─── 指数表示，指数部 4 桁と 5 桁
PRINT USING "_###.###__"; x!────────── #と_を表示
PRINT USING "%####"; x! ────────── %で桁不足を表示
END
```

【結　果】

```
0.123E+02
 1.23E+01
+1.235E+01   +1.235E+001
#12.345_
%   12
```

【例題 28】 書式指定で表を表示

結果のように次のデータを名前を 11 桁 2 桁あけて獲得賞金額を 11 桁で 3 桁ごとのカンマ付き，1 桁あけて年齢を 3 桁で表示するプログラムをつくれ．

データ；名前・獲得賞金・年齢の項とする

```
ｴﾀﾞ ｷﾐｺ,68123415,36,ﾀﾅｶ ﾐﾁﾖ,456365,29
ｲｲｼﾞﾏ ｶｽﾞｺ,712568,33,ｳｴﾀﾞ ﾉﾘｺ,5328756,31
ﾃｼﾞﾏ ｷﾖﾐ,9563255,34,ﾅｶﾑﾗ ｻﾁｺ,658599,28
```

【解　説】 ◎ 書式指定で表を表示

① 文字列，数値などを表示するとき書式指定をしないと，表示開始桁位置や最後の桁がばらばらで表になりません．

② 今回は文字を 11 桁，2 スペースあけて数値を 11 桁で 3 桁ごとのカンマ付き，1 スペースあけて 3 桁で表示します．

【プログラム例】

```
FOR i = 1 TO 6 ─────────────────────────── データを読んで表示
    READ a$, x&, y%
    PRINT USING "&          &  #######,### ###"; a$; x&; y%
NEXT i
END
DATA ｴﾀﾞ ｷﾐｺ,68123415,36,ﾀﾅｶ ﾐﾁﾖ,456365,29
DATA ｲｲｼﾞﾏ ｶｽﾞｺ,712568,33,ｳｴﾀﾞ ﾉﾘｺ,5328756,31
DATA ﾃｼﾞﾏ ｷﾖﾐ,9563255,34,ﾅｶﾑﾗ ｻﾁｺ,658599,28
```

【結　果】

```
ｴﾀﾞ ｷﾐｺ        68,123,415  36
ﾀﾅｶ ﾐﾁﾖ           456,365  29
ｲｲｼﾞﾏ ｶｽﾞｺ        712,568  33
ｳｴﾀﾞ ﾉﾘｺ        5,328,756  31
ﾃｼﾞﾏ ｷﾖﾐ        9,563,255  34
ﾅｶﾑﾗ ｻﾁｺ          658,599  28
```

〔演習問題〕

(27)　a$を"Quick"，b$を"Basic"としてa$を 8 桁，b$を 8 桁で続けて表示し，次にa$とb$の先頭 1 文字を続けて表示するプログラムをつくれ．

(28)　x%を 3568，y!を 152.124 として，①x%を 5 桁，y!を 7 桁，小数点以下 2 桁で，②x%を 6 桁小数点以下 1 桁，y!を 6 桁小数点以下 1 桁で，③x%を 5 桁で+を先頭につけて，y!を 9 桁で小数点以下 3 桁で+を最後につけて表示するプログラムをつくれ．

(29)　x!を 13.259，y!を 22.458 としてz!をその積とし，式と結果を表示するプログラムをつくれ．ただし，①x!とy!は全体 6 桁，小数点以下 3 桁，結果は全体 7 桁，小数点以下 3 桁，②全体を指数で表示するものとする．

(30)　次のデータを a%, b%, c%によんで書式指定を使って，1989　ネン　5　ガツ　15　ニチと表示するプログラムをつくれ．

データ：1989, 5, 15

(31)　次のデータを表にするプログラムをつくれ．項目名として"ショウヒン"，"タンカ"，"コスウ"，"キンガク"をつけ，金額には先頭に"¥"，個数には後に"コ"をつけるものとする．

```
データ：ﾊﾟｿｺﾝ,125000,25,ﾜｰﾌﾟﾛ,73500,110
        ﾌﾟﾘﾝﾀ,68700,25,ﾃﾞｨｽﾌﾟﾚｲ,98500,35
        ﾌﾟﾛｯﾀ,146000,58,ﾌﾛｯﾋﾟｲ,145,3560
```

〔解　答〕

(27)【プログラム例】

```
a$ = "Quick"
b$ = "Basic"
PRINT USING "&       &&       &"; a$; b$ ――― 表示
PRINT USING "!"; a$; b$ ――― 先頭文字の表示
END
```

【結　果】

```
Quick   Basic
QB
```

(28)【プログラム例】

```
x% = 3568
y! = 152.124
PRINT USING "#####  ####.##"; x%; y! ――― 5桁と小数点以下2桁の表示
PRINT USING "####.#  ####.#"; x%; y! ――― 小数点以下1桁の表示
PRINT USING "+####  ####.###+"; x%; y! ――― +付き表示
END
```

【結　果】

```
 3568   152.12
3568.0   152.1
+3568   152.124+
```

(29)【プログラム例】

```
x! = 13.259 ――― 代入
y! = 22.458 ――― 代入
z! = x! * y! ――― 積
PRINT USING "##.###*##.###=###.###"; x!; y!; z! ――― 表示
PRINT USING "##.###^^^^*##.###^^^^=##.###^^^^"; x!; y!; z! ――― 指数表示
END
```

【結　果】

```
13.259*22.458=297.771
 1.326E+01* 2.246E+01= 2.978E+02
```

(30)　【プログラム例】

```
READ a%, b%, c%
PRINT USING "#### ﾈﾝ # ｶﾞﾂ ## ﾆﾁ"; a%; b%; c% ──── 表示
END
DATA 1989,5,15
```

【結　果】

```
1989 ﾈﾝ 5 ｶﾞﾂ 15 ﾆﾁ
```

(31)　【プログラム例】

```
PRINT USING "&          &        &      &   &     &        &        &"; "ｼｮｳﾋﾝ"; "ﾀﾝｶ"; "ｺｽｳ"; "ｷﾝｶﾞｸ"
FOR i = 1 TO 6 ─────────────────────────── 表題
    READ a$, x!, y%                           データを読んで
    z& = x! * y%                              表示
    PRINT USING "&          &  ¥¥###,###  #### ｺ  ¥¥#######,###"; a$; x!; y%; z&
NEXT i
END
DATA ﾊﾟｿｺﾝ,125000,25,ﾜｰﾌﾟﾛ,73500,110
DATA ﾌﾟﾘﾝﾀ,68700,25,ﾃﾞｨｽﾌﾟﾚｲ,98500,35
DATA ﾌﾟﾛｯﾀ,146000,58,ﾌﾛｯﾋﾟｲ,145,3560
```

【結　果】

```
ｼｮｳﾋﾝ            ﾀﾝｶ      ｺｽｳ         ｷﾝｶﾞｸ
ﾊﾟｿｺﾝ       ¥125,000    25 ｺ    ¥3,125,000
ﾜｰﾌﾟﾛ        ¥73,500   110 ｺ    ¥8,085,000
ﾌﾟﾘﾝﾀ        ¥68,700    25 ｺ    ¥1,717,500
ﾃﾞｨｽﾌﾟﾚｲ     ¥98,500    35 ｺ    ¥3,447,500
ﾌﾟﾛｯﾀ       ¥146,000    58 ｺ    ¥8,468,000
ﾌﾛｯﾋﾟｲ          ¥145  3560 ｺ      ¥516,200
```

5章　くり返し

FOR～NEXT, WHILE～WEND, DO～LOOP WHILE, DO～LOOP UNTIL, DO WHILE～LOOP, DO UNTIL～LOOP, EXIT FOR, EXIT DO

【例題29】　FOR～NEXTによる回数指定のくり返し

くり返しを使ってABC…Zを表示するプログラムをつくれ.

【解　説】　◎　FOR～NEXTによる回数指定のくり返しをします.

① 次の書式で指定回数のくり返しを行います.

```
FOR 変数=初期値 TO 終値 STEP 増分
処理
NEXT 変数
```

② 例としてFOR i=5　TO　82　STEP　7　～　NEXT iとするとまずiの値を5とし，処理を行います．次にiに7を加えiの値は12となります．終値に達していなければ再び処理を行います．同様にしてiの値を5，12，19，26，33，40，47，54，61，68，75，82，89となります．iの値が89と終値をこえるとくり返しを終了します.

③ 初期値>終値のときは増分をマイナスとすれば変数の値を毎回減少させてくり返しを行います．たとえば，FOR i=30　TO　10　STEP　-5　処理　NEXT iではiの値を30，25，20，15，10としてくり返しをします．この場合はiの値が5となり，終値より小さくなるとくり返しを終了します.

④ NEXT iの部分はFORの中で使っている変数名と同じものまたは省略します．したがって，例ではNEXT iまたはNEXTです.

⑤ FOR i=500　TO　1000　STEP　-50のように論理が成立しないときは，1回もくり返しは行われません.

⑥ 本題ではABC…Zの表示ですからiの値に65，66，67…90をつくって，そのアスキーコードを表示すればよいことになります．この例のように変数の値を処理の中で使うこともできますし，使わなくてもかまいません.

【プログラム例】

```
FOR i = 65 TO 90 ─────┬─ iが65から90までくり返し
    PRINT CHR$(i); ───┼─ iをアスキーコードとする文字を表示
NEXT i ───────────────┘
END
```

【結　果】

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
```

【例題30】　WHILE〜WEND によるくり返し

整数を入力し，順に表示するプログラムをつくれ．ただし，0 が入力されたら 0 を表示して終わるものとする．

【解　説】　◎　WHILE〜WEND によるくり返しをします．

① WHILE 条件〜WEND は条件が成立している間〜の処理を行います．

② したがって，例解では WHILE x すなわち x が真の間くり返します．くり返しの内容は x の入力とその表示です．0 以外の入力のときはそれをくり返し，0 が入力されると 0 を表示します．次に条件が偽となりますから終了です．

③ 最初に x=1 としておかないと WHILE x が偽となって処理を 1 回もせずに終わります．

④ WHILE　1　〜　WEND では条件が永久に真ですから，無限のくり返しとなります．後でみるように，途中から if 文などを使ってぬけでることができます．

【プログラム例】

```
x = 1 ──────────── x を 1 とする．次の WHILE 条件を真とする目的
WHILE x ────────┬─ x が真の間くり返し
    INPUT x     │
    PRINT x     │
WEND ───────────┘
END
```

【結　果】

```
? 5                        5, 2, 45, -87, 26, 0 を入力の例
 5
? 2
 2
? 45
 45
? -87
-87
? 26
 26
? 0
 0
```

【例題31】　DO〜LOOP WHILE によるくり返し

次のデータを読んで 0 まで表示するプログラムをつくれ．

データ;5, 2, 4, 8, 2, 9, 0, 1, 4

【解　説】　◎　DO〜LOOP WHILE によるくり返しをします．

① Quick BASIC は DO〜LOOP という汎用くり返しを準備しています．これは 4 つの変形を持っています．

	くり返しの条件	条件判断のタイミング
DO～LOOP WHILE　条件	……条件が成立している間	処理の後
DO～LOOP UNTIL　条件	……条件が成立するまで	処理の後
DO　WHILE 条件～LOOP	……条件が成立している間	処理の前
DO　UNTIL 条件～LOOP	……条件が成立するまで	処理の前

② 本例は DO　処理　WHILE x<>0 です．処理を行ってから，x が 0 でなければくり返します．データが 0 の場合は x に 0 を読み，x を表示した後，x が 0 となり条件が偽となって終了します．

【プログラム例】

```
DO ──────────────── くり返し
    READ x ──────── データを読んで表示
    PRINT x;
LOOP WHILE x <> 0 ── x が 0 でないときくり返し
END
DATA 5,2,4,8,2,9,0,1,4
```

【結　果】

```
5  2  4  8  2  9  0
```

【例題 32】　DO～LOOP UNTIL

数 x を入力して，x の階乗を求めるプログラムをつくれ．

【解　説】　◎　DO～LOOP UNTIL のくり返しをします．

① DO～LOOP UNTIL　条件では条件が成立するまでくり返しをします．条件の判断は処理をすませた後です．

② 本例での t，y の動きは次のようになります．

	t（階乗の値）	y	
初期値	1	5（入力した値）	
1回目	5　　(1 * 5)	4 (5－1)	
2回目	20　(5 * 4)	3 (4－1)	
3回目	60　(20 * 3)	2 (3－1)	
4回目	120 (60 * 2)	1 (2－1)	
5回目	120 (120 * 1)	0 (1－1)	y の値が 0 となってくり返しを終了

③ このプログラムでは x に 0 を入力すると y=y-1 で y の値は-1 となり，次は-2，-3，…となります．したがって，y が 0 になりませんから無限にくり返しが続きます．積 t=t*y が最初に t=0 となるため t の値は 0 です．

④ また，x に実数で小数点のつく値をいれると y の値が 0 になりませんからこれも無限にくり返されます．ただし，この場合は t の値はオーバーフローして終了します．たとえば，x に 5.3 を入力すると t の値は 1*5.3*4.3*3.3*2.3*1.3*0.3*-0.7*-1.7*-2.7*…となります．

【プログラム例】

```
t = 1 ──────────────── 階乗の入る変数の初期値
INPUT x ────────────── x の入力
y = x ──────────────── x の値を y に代入
DO ──────────────┐──── くり返し
    t = t * y    │     積 1*y*(y-1)*(y-2)…
    y = y - 1    │     y-1, y-2, y-3, …をつくる
LOOP UNTIL y = 0 ┘     y が 0 になるとくり返しを終了
PRINT x; "!="; t ───── 結果の表示
END
```

【結　果】

```
? 10                          10 を入力の例
 10 != 3628800
```

【例題 33】　DO UNTIL～LOOP

a，b，c さんの成績は各々60，80，50 点とする．次に最低点 x を入力し，x が 40 点未満のとき，1 回に 10 点のげたをはかせる．この時 1 回につき a は 3 点，b は 2 点，c は 4 点のげたを上のせするものとする．最低点がげたを何回かはいて 40 点以上となったときの x，a，b，c の点を表示するプログラムをつくれ．

【解　説】　◎　DO UNTIL～LOOP によるくり返しをします．

① DO UNTIL　条件～LOOP では条件が成立するまで処理をくり返します．

② 処理の結果で条件を判断する場合は DO UNTIL～LOOP も DO～LOOP UNTIL も同じです．ただし，DO～LOOP へ入る前の値で条件が決まる（本例は入力された x の値で条件が決まる）ときはこの 2 つは意味が違います．

x=10	x=10
DO UNTIL x>=5	DO
処理	処理
LOOP	LOOP UNTIL x>=5
x が 10 なので x>=5 の条件が成立するのでループ内の処理は 1 回もしないで LOOP の次へ進む	x が 10 でも処理を 1 回行って x>=5 が成立するので LOOP の次へ進む

③ x の点を 25 とすると次のような処理となります．

	初期値	1 回目の処理	2 回目の処理	
a	60	63	66	
b	80	82	84	
c	50	54	58	
x	25	35	45	（x>=40 のため終了）

【プログラム例】

```
a = 60 ─────────────┬─a, b, c の初期値
b = 80              │
c = 50 ─────────────┘
INPUT x ─────────────── x の入力
DO UNTIL x >= 40 ───┬── x>=40 になるまでくり返し
    x = x + 10 ─────┼── x に 10 を加える
    a = a + 3 ──────┼── a に 3 を加える
    b = b + 2 ──────┼── b に 2 を加える
    c = c + 4 ──────┼── c に 4 を加える
LOOP ───────────────┘
PRINT "x="; x, "a="; a ─┬─ x, a, b, c の表示
PRINT "b="; b, "c="; c ─┘
END
```

【結　果】

```
? 25                              x=25 を入力の例
x= 45           a= 66
b= 84           c= 58

? 50                              x=50 を入力の例
x= 50           a= 60
b= 80           c= 50

? 35                              x=35 を入力の例
x= 45           a= 63
b= 82           c= 54
```

【例題 34】　DO WHILE～LOOP

数 x を入力して 2^x を求めるプログラムをつくれ.

【解　説】　◎　DO WHILE～LOOP によるくり返しをします.

① DO WHILE　条件～LOOP は処理の前に条件を判断し，条件が成立しているとき処理を行い，くり返しをします．条件が成立しないときは LOOP の外へ出ます．したがって，最初に条件が成立しないと，処理は一度も実行されません.

② 本例では x に整数を入力し，その回数だけ 2 を掛けます．整数から順に 1 をひいていって，値が 0 になるとくり返しの終了です.

③ x に 0 が入力されると条件が不成立のため，処理はされずもとの z%=1 が生きていて，2^0 が 1 となります.

【プログラム例】

```
z% = 1 ──────────────────── 積の入る変数の初期値
INPUT x% ────────────────── x の入力
y% = x% ─────────────────── y に x を代入
DO WHILE y% <> 0 ──────┬─── くり返し．y が 0 でないときくり返す
    z% = z% * 2 ─────┬─┼─── 2^x を求める
    y% = y% - 1 ─────┘ │
LOOP ──────────────────┘
PRINT "2^"; x%; "="; z% ─── 結果の表示
END
```

【結　果】

```
? 6                    6を入力の例
2^ 6 = 64

? 0                    0を入力の例
2^ 0 = 1
```

【例題35】　FOR〜NEXTのネスティング

上段に横に123…10とかき，縦に123…10とかき，その交点に横の値と縦の値の積をかく10*10の表をつくるプログラムをつくれ．

```
1    2    3    4……10
2    4    6    8    20
3    ↑         12←4と3の積
     2と2の積
4
⋮
                  ┌── 10と10の積
                  ↓
10  20  30  40   100
```

【解　説】　◎　FOR〜NEXT文のネスティングをします．

① 次の形で2重のくり返し（ネスティング）をします．

```
FOR i=1  TO  10 ─────────────┐ iを1から10までくり返す
     FOR j=1  TO  10 ───┐    │ iを1としjを1, 2, 3, …10
                        │ jを1から10までくり返す
                        │    │ iを2としjを1, 2, 3, …10
        処　理          │    │ iを3としjを1, 2, 3, …10
                        │    │    ⋮
     NEXT j ────────────┘    │
NEXT i ──────────────────────┘ iを10としjを1, 2, 3, …10とする
```

② ネスティングは何重になってもかまいません．ただし，中のFOR〜NEXT文は外のFOR〜NEXTの中に完全に入っていなければなりません．

正しい例

```
FOR i=1  TO  5 ─────────────────┐
     FOR j=5  TO  50 ─────────┐ │
          FOR k=20  TO  100 ┐ │ │
          処　理            │ │ │
          NEXT k ───────────┘ │ │
     NEXT j ──────────────────┘ │
NEXT i ─────────────────────────┘
```

正しくない例

```
┌─ FOR i=1  TO  5
│  ┌─── FOR j=5  TO  50
│  │ ┌───── FOR k=20  TO  100
│  │ │      処　理
│  └─┼───── NEXT j
│    └─── NEXT k
└─ NEXT i
```

③　FOR〜NEXT 文がネスティングされ，NEXT が並ぶような場合は次のようにすることができます．下の３例はいずれも正しい使い方です．

```
FOR i=1  TO  10
    FOR j=3  TO  8
    処 理
    NEXT j
NEXT i
```

```
FOR i=1  TO  10
    FOR j=3  TO  8
    処 理
    NEXT
NEXT
```

```
FOR i=1  TO  10
    FOR j=3  TO  8
    処 理
NEXT j, i
```

【プログラム例】

```
FOR i = 1 TO 10 ──────────── i=1 から 10 までくり返し
   FOR j = 1 TO 10 ──────── j=1 から 10 までくり返し
      PRINT USING "####"; i * j; ── i*j の表示
   NEXT j
   PRINT ──────────── 改行
NEXT i
END
```

1*1, 1*2, 1*3, …1*10
2*1, 2*2, 2*3, …2*10
⋮
10*1, 10*2, 10*3, …10*10 を表示

【結　果】

```
   1   2   3   4   5   6   7   8   9  10
   2   4   6   8  10  12  14  16  18  20
   3   6   9  12  15  18  21  24  27  30
   4   8  12  16  20  24  28  32  36  40
   5  10  15  20  25  30  35  40  45  50
   6  12  18  24  30  36  42  48  54  60
   7  14  21  28  35  42  49  56  63  70
   8  16  24  32  40  48  56  64  72  80
   9  18  27  36  45  54  63  72  81  90
  10  20  30  40  50  60  70  80  90 100
```

【例題 36】　FOR〜NEXT からの脱出

1+2+3+…+i が 1000 をこえる最初の i の値とそのときの和を求めるプログラムをつくれ．

【解　説】　◎　EXIT FOR は FOR〜NEXT ループの途中で脱出します．

①　FOR i=初期値　TO　終値〜NEXT 文の中で i が終値になる前に別の条件でくり返しを終わらせたいことがあります．そのために IF　条件　THEN EXIT FOR という形が使えます．EXIT FOR は NEXT 文の次へ進みます．

②　本例では 1+2+…の和が t に求められます．この和 t が 1000 をこえたとき EXIT FOR でくり返しから脱出します．

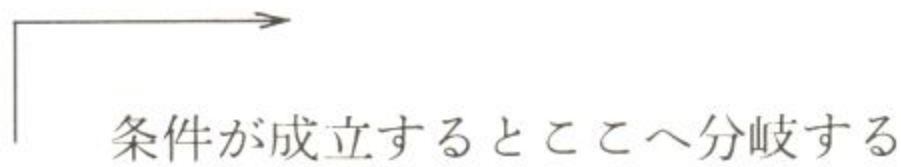
③　i の終値は充分大きくして IF 文の条件によるループの脱出より先にくり返しが終わらないように

します.

④　従来の BASIC では GOTO 文あるいは GOSUB 文でループの外へ脱出しました.

新しい形

```
FOR i=1  TO  100
        処 理
        IF  条件  THEN EXIT FOR
NEXT i
```

条件が成立するとここへ分岐する

従来の形

```
 10  FOR i=1  TO  100
  ~        処 理
 80        IF  条件  THEN GOTO  100
 90  NEXT i
100
```

条件が成立するとここへ分岐する

【プログラム例】

```
t = 0 ──────────────────── 和の入る変数 t の初期値を 0 とする
FOR i = 1 TO 100 ────────── i を 1 から 100 までくり返し
    t = t + i ───────────── t に 1+2+… を求める
    IF t > 1000 THEN EXIT FOR── t が 1000 をこえると脱出
NEXT i
PRINT "1+2+3+・・・・+"; i; "="; t ── 結果の表示
END
```

【結　果】

```
1+2+3+・・・・+ 45 = 1035
```

【例題 37】　EXIT DO による脱出

数 x を入力して表示するプログラムをつくれ. ただし, x がマイナスのとき, その値を表示しないで終了するものとする.

【解　説】　◎　EXIT DO で DO～LOOP ループの途中から脱出します.

①　DO～LOOP の中に IF　条件　THEN EXIT DO を入れると条件が成立したとき DO～LOOP から脱出できます.

②　本例では次のようになります.

x に 0 以上の値が入力されている間は DO UNTIL x<0 の条件が成立しませんから入力と表示を

くり返します.

x にマイナス値が入力されると IF x<0　THEN EXIT DO によって，終了します．つまり表示をしないで終了します．この IF 文がないと，マイナスの x の値を表示した後 DO UNTIL x<0 の条件が成立して終了します.

この例では IF 文は x がマイナスのとき，その値を表示するかしないかの違いです.

【プログラム例】

```
DO UNTIL x < 0 ──────────┐x がマイナスになるまでくり返し
    INPUT x ─────────────┤x の入力
    IF x < 0 THEN EXIT DO ┤x がマイナスのとき LOOP の次へ脱出
    PRINT x ─────────────┤x の表示
LOOP ────────────────────┘
END
```

【結　果】

```
? 110             110, 25, -86 を入力の例
 110
? 25
 25
? -86
```

【例題38】　EXIT DO による DO～LOOP UNTIL からの脱出

アルファベット大文字 A～J を順に表示するプログラムをつくれ.

【解　説】　◎　EXIT DO によって DO～LOOP から脱出します.

① DO～LOOP の処理の中に IF　条件　THEN EXIT DO を入れると条件が成立したとき DO～LOOP の外へ脱出できます.

② 本例では DO～LOOP 条件の条件は A, B, C…と表示していって Z になったときです．しかし，途中で IF t>=75　THEN EXIT DO とします．すなわち，t が 75（J のアスキーコード）のとき EXIT DO でループの外へ脱出します．すなわち，A～J を表示して終了します.

【プログラム例】

```
t = 64 ────────────────────┐t の初期値を 64 とする
DO ────────────────────────┤くり返し，x$が"z"で終了
    t = t + 1 ─────────────┤t に 1 を加え 65, 66, 67…をつくる，A, B, C…のアスキーコード
    IF t >= 75 THEN EXIT DO ┤t が 75 以上になったらループの外へ脱出
    PRINT CHR$(t); ─────────┤表示
LOOP UNTIL x$ = "Z" ────────┘
END
```

【結　果】

```
ABCDEFGHIJ
```

〔演習問題〕

(32)　アルファベット小文字で z から a まで 1 行に表示するプログラムをつくれ.

(33)　"アイウエオカキクケコ"を 1 桁おきに 1 行に表示するプログラムをつくれ.

(34)　WHILE〜WEND を使って"ABC…Z"を表示するプログラムをつくれ.

(35)　1 桁の整数の乱数をつくり表示するプログラムをつくれ. ただし, 0 のとき表示して終了するものとする.

(36)　下のデータを読み, 1 行に表示するプログラムをつくれ. ただし, "E"のとき, それを表示して終了するものとする.

データ;T, O, K, Y, O, E, H, I, M, E

(37)　$1^2+2^2+3^2+\cdots i^2$ が 5000 をこえる最初の和を求めるプログラムをつくれ.

(38)　下のデータを読み表示するプログラムをつくれ. ただし, 0 の前までのデータとする.

データ;5, 2, 3, 7, 8, 1, 2, 3, 9, 1, 3, 8, 1, 7, 1,
0, 5, 2, 9, 9

(39)　DO　UNTIL〜LOOP を使い, x にマイナスの値が入力されるまで値を入力し, 表示をくり返すプログラムをつくれ.

(40)　数 x を入力し, x^2 を表示するプログラムをつくれ. x^2 が 10000 以上となったとき終了するものとする.

(41)　1000 から下のデータを順にマイナスして, 0 以下となったときの値を求めるプログラムをつくれ.

データ;120, 52, 456, 85, 92, 140, 358

(42)　1+2+3+…+t を行い和が 10000 をこえる最小の t の値と和を求めるプログラムをつくれ.

〔解　答〕

(32) **【プログラム例】**

```
FOR i = 122 TO 97 STEP -1 ─┐ i が 122 から 97 までくり返し
    PRINT CHR$(i);  ────────┤ i の値をアスキーコードとする文字を表示
NEXT i ─────────────────────┘
END
```

【結　果】

zyxwvutsrqponmlkjihgfedcba

(33) **【プログラム例】**

```
FOR i = 177 TO 186 ──────┐ i が 177 から 186 までくり返し
    PRINT CHR$(i); " "; ──┤ i の値をアスキーコードとする文字を表示
NEXT i ───────────────────┘
END
```

【結　果】

アイウエオカキクケコ

(34) 【プログラム例】

```
x = 1 ―――――――― くり返しの条件を成立させるため x=1 とする
y = 65 ――――――― y の初期値"A"のアスキーコード
WHILE x ―――――― x が 0 以外のときくり返し
    PRINT CHR$(y); ― y をアスキーコードとする文字の表示
    y = y + 1 ――― y を 65, 66, 67, …91 とする
    x = 91 - y ―― x を 91-y とする. y が 90 で z を表示した後,
WEND ―――――――   y=y+1 により y が 91 となり, x=91-y により
END                x が 0 となって終了する
```

【結　果】

ABCDEFGHIJKLMNOPQRSTUVWXYZ

(35) 【プログラム例】

```
x = 1 ―――――――――― くり返しの条件を成立させるため x=1 とする
WHILE x ―――――――― x が 0 以外のときくり返し, 乱数の値が 0 となると終了
    x = INT(RND * 10) ― x に 1 桁の乱数を代入
    PRINT x; ――――― x の表示
WEND ―――――――――
END
```

【結　果】

7　5　5　2　3　7　0

(36) 【プログラム例】

```
x = 1 ―――――――――― くり返しの条件を成立させるため x=1 とする
WHILE x ―――――――― x が 0 以外のときくり返し, x$が"E"のとき x が 0 で終了
    READ x$ ―――――― 1 文字ずつ読み x$に代入
    PRINT x$; ――――― x$の表示
    x = ASC(x$) - 69 ― 文字のアスキーコード－69 を x に代入. "E"のアスキーコ
WEND ―――――――――   ードが 69 のため"E"のとき x が 0 となる.
END
DATA T,O,K,Y,O,E,H,I,M,E
```

【結　果】

TOKYOE

(37) 【プログラム例】

```
t = 0 ――――――――――――――― i の初期値を 0 とする. i は和が入る
FOR i = 1 TO 100 ――――――――― i を 1 から 100 までくり返す
    t = t + i * i ―――――――― i² の和を求める
    IF t > 5000 THEN EXIT FOR ―― 和が 5000 をこえたら終了
NEXT i ―――――――――――――
PRINT "1^2+2^2+3^2+・・・・+"; i; "^2="; t ―― 結果の表示
END
```

【結　果】

```
1^2+2^2+3^2+・・・・+ 25 ^2= 5525
```

(38) 【プログラム例】

```
DIM a(20)                                  データの読み込み
FOR i = 1 TO 20 ─────────────┬─ 20 個分読む準備をする
    READ a(i) ───────────────┼─ データの読み込み
    IF a(i) = 0 THEN EXIT FOR ─┼─ データが 0 のときループ終了
NEXT i ──────────────────────┘
FOR i = 1 TO 20 ─────────────┬─ 20 個分表示する準備をする
    IF a(i) = 0 THEN EXIT FOR ─┼─ データが 0 のとき終了
    PRINT a(i); ─────────────┼─ データの表示
NEXT i ──────────────────────┘
END
DATA 5,2,3,7,8,1,2,3,9,1
DATA 3,8,1,7,1,0,5,2,9,9
```

【結　果】

```
5  2  3  7  8  1  2  3  9  1  3  8  1  7  1
```

(39) 【プログラム例】

```
DO UNTIL x < 0 ─────┬─ x がマイナスになるまでくり返し
    INPUT x ────────┼─ x を入力
    PRINT x ────────┼─ x の表示
LOOP ───────────────┘
END
```

【結　果】

```
? 150                  150, 0, 25, -54 を入力の例
 150
? 0
 0
? 25
 25
? -54
-54
```

(40) 【プログラム例】

```
DO ──────────────────────────┬─ z が 10000 以下のときくり返し
    INPUT x ─────────────────┼─ x を入力
    z = x * x ───────────────┼─ z に x² を代入
    PRINT x; "^2"; "="; z ───┼─ x² を表示
LOOP WHILE z < 10000 ────────┘
END
```

【結　果】

```
? 55                               55, 96, 150 を入力の例
 55 ^2= 3025
? 96
 96 ^2= 9216
? 150
 150 ^2= 22500
```

(41) 【プログラム例】

```
t = 1000 ─────────────── tの初期値
DO ───────────────────── tが0以下になるまでくり返し
    READ x ───────────── xにデータを読み込む
    t = t - x ────────── tからxを順にひいていく
LOOP UNTIL t <= 0
PRINT t ──────────────── 結果の表示
END
DATA 120,52,456,85,92,140,358
```

【結 果】

```
-303
```

(42) 【プログラム例】

```
t = z = 0 ────────────── tとzの初期値，tは1，2，3，…をつくる．zは1+2+3…の和
DO ───────────────────── zが10000以上になるまでくり返し
    t = t + 1 ────────── tを1，2，3…とする
    z = z + t ────────── zに1，2，3…を求める
LOOP UNTIL z >= 10000
PRINT t; z ───────────── 結果の表示
END
```

【結 果】

```
141   10011
```

6章 条件分岐

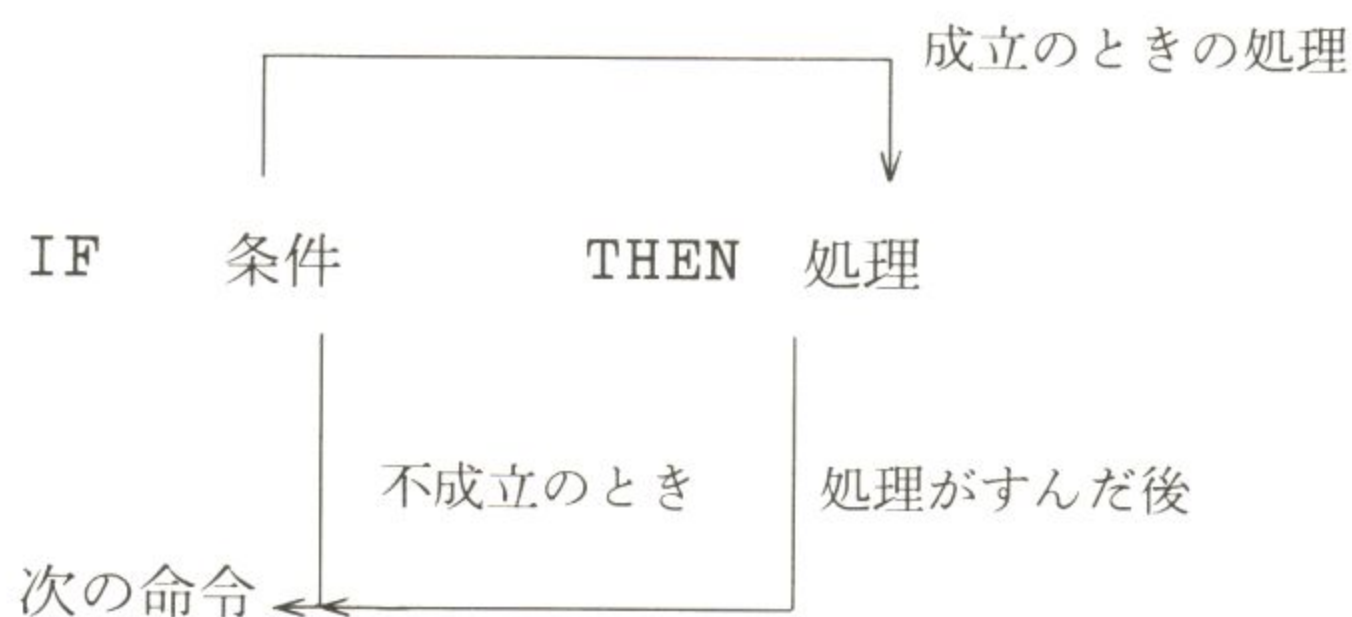
IF〜THEN, IF〜THEN〜ELSE, ブロック IF,
END IF, ELSEIF, OR, AND, XOR, EQV, IMP, NOT

【例題 39】 IF〜THEN

文字を入力し，"A"または先頭が"A"の文字列のとき表示するプログラムをつくれ.

【解 説】 ◎ IF〜THEN〜の分岐をします.

① IF 条件 THEN 処理によって条件が成立すると処理を行い条件が不成立のときは次へ進みます.

```
      ┌─────────────────┐ 成立のときの処理
      │                 ↓
IF    条件        THEN  処理
        │              │
        │  不成立のとき │ 処理がすんだ後
次の命令 ←──────────────┘
```

② 本例では入力された文字のアスキーコードが 65 すなわち"A"のとき，その文字を表示します. 65 でないときと，表示がすむと終了です.

【プログラム例】

```
INPUT x$ ──────────────────── 文字の入力
IF ASC(x$) = 65 THEN PRINT x$── 先頭が"A"のとき表示
END
```

【結 果】

```
? Tokyo                    Tokyo を入力の例

? Akasaka                  Akasaka を入力の例
Akasaka
```

【例題 40】 IF〜THEN〜ELSE〜

数を入力し，100 以上のとき">=100"と表示し，それ以外では"<100"と表示するプログラムをつくれ.

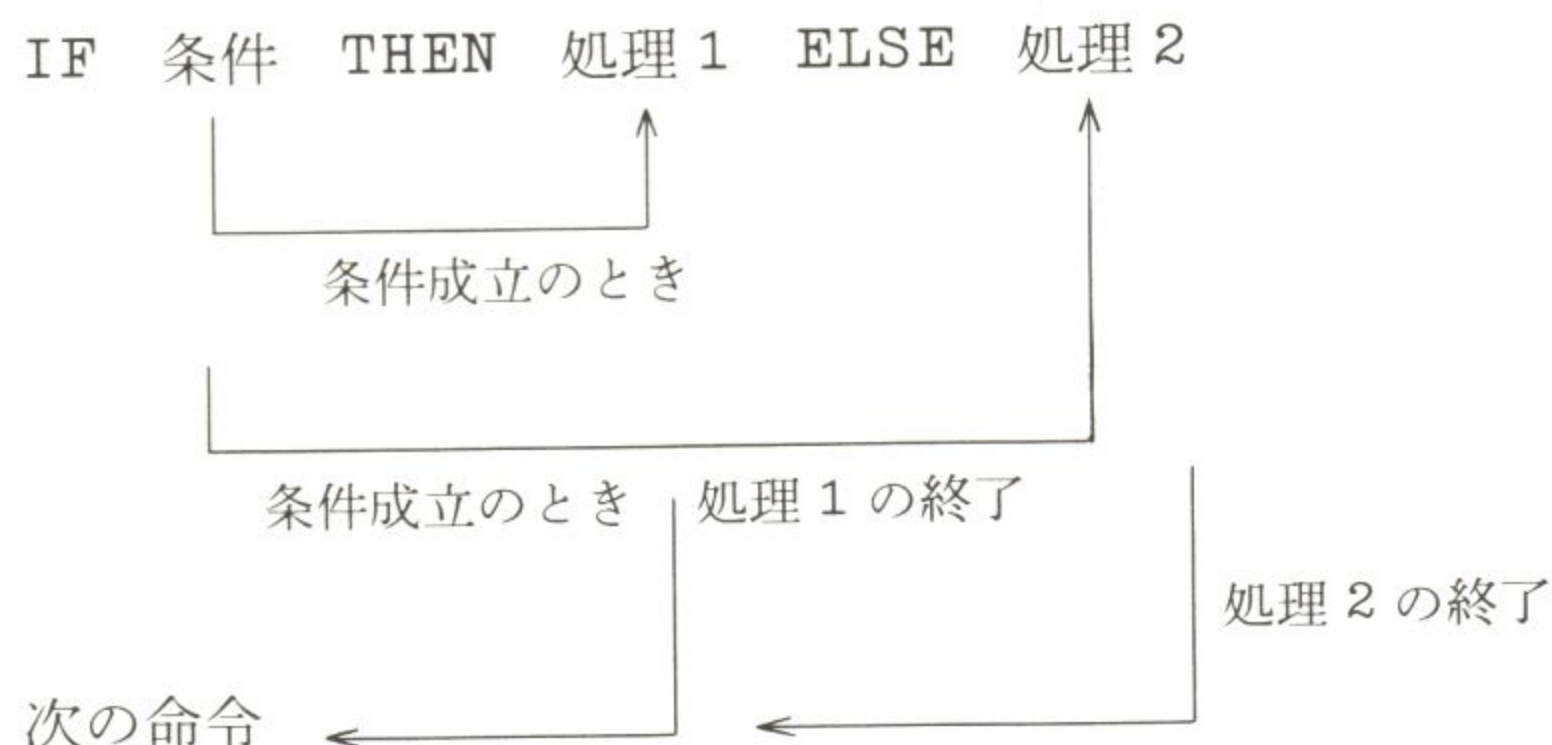
【解 説】 ◎ IF 条件 THEN〜ELSE〜の分岐をします.

① 次の形で条件によって分岐します.

```
IF 条件 THEN 処理1 ELSE 処理2
```

条件成立のとき

条件成立のとき　処理1の終了

処理2の終了

次の命令

② 本例ではxに値を入力し, 100以上のときは">=100"と表示し, 100未満のときは"<100"と表示します. どちらも処理が終わると次の命令ENDで終了します.

【プログラム例】

```
INPUT x ──────────────────────────── 数値の入力
IF x >= 100 THEN PRINT ">=100" ELSE PRINT "<100"──── 2条件に分けて表示
END
```

【結　果】

```
? 1250 ──────── 1250を入力の例
>=100

? 10             10を入力の例
<100
```

【例題41】　ブロックIF～THEN～ELSE

数xを入力し, 正のときはyにxを代入し, 負のときはyに-xを代入して$\sqrt{y}$を求めて表示するプログラムをつくれ.

【解　説】　◎　ブロックIF～THEN～ELSEによる分岐をします.

① ブロックIFは次の形で分岐をします.

```
IF 条件 THEN

  処理1 (命令群)    …条件が成立したとき行う処理
                     複数の命令をかくことができる

ELSE

  処理2 (命令群)    …条件が不成立のとき行う処理
                     複数の命令をかくことができる

END IF
```

処理1, 処理2の内容が複雑な場合, 簡潔に表現できます.

② 処理1の内容が命令1, 命令2, …命令5, 処理2の内容が命令6, 命令7, …命令9でできているとします. ブロックIFを使うと次のようにかけます. 従来のBASICと比較してください.

```
　　　　ブロック IF　　　　　　従来の BASIC 例①
IF　　条件　THEN　　　　　IF　条件　THEN　命令１：命令２：命令３：命令４：
　　　　　　　　　　　　　　　　　　　　　　命令５　ELSE　命令６：命令７：
　　　　　　　　　　　　　　　　　　　　　　命令８：命令９
　　　　命令１　　　　　　次の命令
　　　　命令２　　　　　　　従来の BASIC 例②（×××，○○○，△△△は行番号）
　　　　命令３　　　　　　　IF　条件　THEN　×××　ELSE　○○○ ―― 一度×××と
　　　　　　　　　　　　　　　　　　　　　　○○○に分けて処理先を示して分岐させる
　　　　命令４　　×××　　　命令１
　　　　命令５　　　　　　　命令２
ELSE　　　　　　　　　　　　命令３
　　　　命令６　　　　　　　命令４
　　　　命令７　　　　　　　命令５
　　　　命令８　　　　　　　GOTO　△△△ ―― 処理がすむと GOTO 文で次の
　　　　命令９　　○○○　　　命令６　　　　　　命令の行へとばす
END IF　　　　　　　　　　　命令７
次の命令　　　　　　　　　　命令８
　　　　　　　　　　　　　　命令９
　　　　　　　　　△△△　　次の命令
```

③　本例では処理１，処理２の内容が各々命令１つですからブロック IF を使わなくても簡単にかけます．ブロック IF があるからといってすべてブロック IF を使う必要はありません．

　IF x>=0　THEN y=x ELSE y=-x　（ブロック IF を使わない例）

【プログラム例】

```
INPUT x ―――――――――― x の入力
IF x >= 0 THEN ――――┐ x が正のとき
     y = x ――――――――┘
ELSE ――――――――――――――― x が負のとき
     y = -x ―――――――┘
END IF
y = SQR(y) ―――――――┐ √y の表示
PRINT y ――――――――――┘
END
```

【結　果】

```
? 135              135 を入力の例
 11.61895
? -547             -547 を入力の例
 23.38803
```

【例題 42】　ブロック IF～THEN

次のデータを使い最も文字数の長い都市名を表示するプログラムをつくれ．

```
Chicago,Paris,Rome,London
Boston,Tokyo,Pittsburgh,Bon
```

【解　説】　◎　ブロック IF～THEN を使って最大値を求めます．

① ブロック IF　条件　THEN　処理　END IF を使います．ELSE はありません．

② 最大値を求める方法は，前から順に都市名を読みその文字数と max を比較します．最初に max=0 としておき，文字数が max より大きければその値を次の max とします．順にこれをくり返すと max に最大値が入ります．

③ 具体的にみてみましょう．

	1 回目	2 回目	3 回目
max 値（最初は 0）	0	7	7
x$	Chicago	Paris	Rome
文字数	7	5	4
max	7	7	7
y$	Chicago	Chicago	Chicago

……7 回目	8 回目
7	10
Pittsburgh	Bon
10	3
10	10
Pittsburgh	Pittsburgh

④
```
IF l>=max THEN
    max=l
    y$=x$
END IF
```
の部分がブロック IF 文です．l>=max のとき新しい l を max に入れ，新しい文字列 x$を y$に代入します．条件が不成立のとき，(新しい都市名がその前の最長文字数の都市名より短いとき）は特に処理はしませんので ELSE はありません．

【プログラム例】

```
max = 0 ──────────────── 最大値が入る
FOR i = 1 TO 8 ─────────── 8 文字列の比較をくり返す
    READ x$ ───────────── 文字列の読み込み
    l = LEN(x$) ─────────── 文字数
    IF l >= max THEN ────── 文字数が大きいとき l を max，y$を x$とする
        max = l
        y$ = x$
    END IF
NEXT i
PRINT y$ ─────────────── y$（最長文字列）の表示
END
DATA Chicago,Paris,Rome,London
DATA Boston,Tokyo,Pittsburgh,Bon
```

【結　果】

Pittsburgh

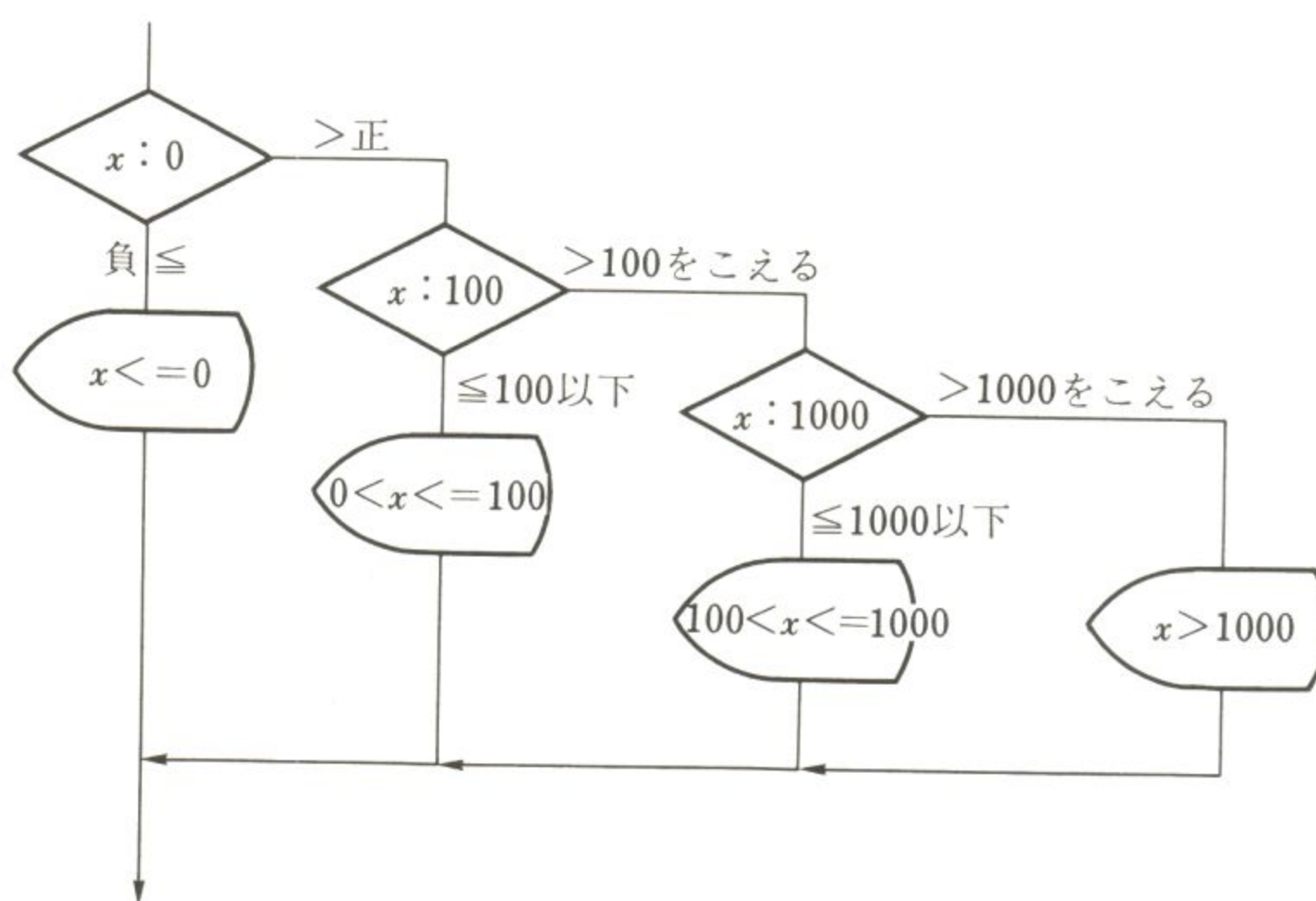
【例題 43】　ブロック IF の多重化

数 x を入力して負のとき”x<=0”，100 以下のとき”0<x<=100”，1000 以下のとき”100<x<=1000”，1000 をこえるとき”x>1000”と表示するプログラムをつくれ．

【解　説】　◎　ブロック IF の多重化をします．

① IF 文の中に IF 文を使うことができます．これを多重化といいます．IF 文は IF～THEN～ELSE と THEN と ELSE を持つもの，IF～THEN と THEN のみのものがありますので，あまり複雑にするとどの IF と THEN，ELSE が対応するのかわからなくなってしまいます．1 つのヒントは対応する THEN と ELSE の字下げの位置を同じにすることが考えられます．

② 本例では次のように考えます．

まず，数値が負か正かに分けます．負の場合は”x<=0”と表示します．正の場合は 100 以下か，100 をこえるかに分けます．100 以下のときは”0<x<=100”と表示します．100 をこえるときは 1000 以下か 1000 をこえるかに分けます．1000 以下のときは”100<x<=1000”と表示し，1000 をこえるときは”x>1000”と表示します．

③ ELSE の中で IF 文を使う場合は ELSEIF を使います．

```
IF  条件  THEN
        処理 1      ―――― 条件 1 が成立するとき行う処理
                         条件 1 が不成立のときは処理 2 か処理 3 へ進む
ELSEIF  条件 2  THEN
            処理 2  ―――― 条件 2 が成立したときの処理
        ELSE
            処理 3  ―――― 条件 2 が不成立のときの処理
END IF
```

【プログラム例】

```
INPUT x ─────────────────────── x の入力
IF x <= 0 THEN ──────────────┐─ x が負のときの表示
    PRINT "x<=0"─────────────┘
ELSEIF x <= 100 THEN─────────┐─ x が正で 100 以下のときの表示
    PRINT "0<x<=100"─────────┘
    ELSEIF x <= 1000 THEN────┐─ x が 100 をこえ 1000 以下のときの表示
        PRINT "100<x<=1000"──┘
        ELSE ────────────────┐─ x が 1000 をこえるときの表示
        PRINT "x>1000"; ─────┘
END IF
END
```

【結　果】

```
? 358                    358 を入力の例
100<x<=1000

? 3590                   3590 を入力の例
x>1000

? -500                   -500 を入力の例
x<=0
```

【例題 44】　OR 条件

下のデータのうち"R"で始まるか，"T"で始まる都市名を表示するプログラムをつくれ.

データ;Chicago, Paris, Rome, London, Boston, Tokyo,
　　Pittsburgh, Bon

【解　説】　◎　IF 条件の条件に OR を使って分岐します.

① IF 文の条件には関係演算子と組合わせとして下の論理演算子が使えます.

関係演算子の種類		論理演算子	
=	等しい	NOT	否定
>	大きい	AND	論理積
<	小さい	OR	論理和
<>	等しくない	XOR	排他的論理和
<=	以下	EQV	同値
>=	以上	IMP	包含

② 今回は OR を使います．OR はどちらかの条件が成立すれば全体の条件が成立します.

IF　条件 1　OR　条件 2　THEN　処理 1　ELSE　処理 2

条件 1 または条件 2 が成立すれば処理 1 を，どちらも不成立なら処理 2 を行います.

③ OR 条件は次のようになります.

X	Y	X OR Y
T	T	T
T	F	T
F	T	T
F	F	F

TはTRUE
FはFALSE

【プログラム例】

```
FOR i = 1 TO 8 ———————————————— 8回のくり返し
    READ x$ ———————————————— x$にデータを読む
    IF ASC(x$) = 82 OR ASC(x$) = 84 THEN PRINT x$ —— x$がRまたはTで始まっているとき表示
NEXT i ————————————————
END
DATA Chicago,Paris,Rome,London
DATA Boston,Tokyo,Pittsburgh,Bon
```

【結　果】

```
Rome
Tokyo
```

【例題45】　AND条件

下のデータのうち6文字以下で最後に"n"を持つ文字列を表示するプログラムをつくれ.

データ;Chicago, Paris, Rome, London, Boston, Tokyo,
Pittsburgh, Bon

【解　説】　◎　AND条件を使って分岐します.

①　IF条件の中にANDを使うと2つの条件が同時に成立したとき分岐します.

```
IF LEN(x$)=<6  AND RIGHT$(x$)="n"  THEN PRINT x$
```

条件1（6文字以下）　同時成立　条件2（最後が"n"）

②　AND条件は次のようになります.

X	Y	X AND Y
T	T	T
T	F	F
F	T	F
F	F	F

TはTRUE
FはFALSE

【プログラム例】

```
FOR i = 1 TO 8
    READ x$
    IF LEN(x$) <= 6 AND RIGHT$(x$, 1) = "n" THEN PRINT x$ —— 文字数が6以下で
NEXT i                                                    最後が"n"のとき表示
```

```
END
DATA Chicago,Paris,Rome,London
DATA Boston,Tokyo,Pittsburgh,Bon
```

【結　果】

```
London
Boston
Bon
```

【例題 46】　NOT 条件

下のデータの中で文字数が 7 以下でない文字列を表示するプログラムをつくれ.

データ；Chicago,Paris,Rome,London,Boston,Tokyo,Pittsburgh,Bon

【解　説】　◎　NOT 条件を使います.

① IF の条件の中に NOT を使うとその条件ではない場合分岐します.

```
IF NOT(LEN(x$)=<7)  THEN PRINT x$
```

文字数が 7 以下ではない

否定

IF LEN(x$)>7　THEN PRINT x$と同じ

② NOT 条件は次のようになります.

X	NOT X
T	F
F	T

T はTRUE
F はFALSE

【プログラム例】

```
FOR i = 1 TO 8
    READ x$
    IF NOT (LEN(x$) <= 7) THEN PRINT x$ ―― 文字数が 7 以下でないとき表示
NEXT i
END
DATA Chicago,Paris,Rome,London
DATA Boston,Tokyo,Pittsburgh,Bon
```

【結　果】

```
Pittsburgh
```

【例題 47】　XOR 条件

次のデータのうち 4 文字以下か，最後に"n"を持つ 2 条件のどちらか 1 つが成立し，他が成立しない文字列を表示するプログラムをつくれ.

データ；Chicago, Paris, Rome, London, Boston, Tokyo, Pittsburgh, Bon

【解　説】 ◎ XOR 条件を使って分岐します.

① XOR は条件 1　XOR　条件 2 で一方の条件が成立し，他方が成立しない場合に真です. したがって，本例では文字数が 4 以下で最後が"n"でない場合か，文字数が 4 をこえて最後が"n"の場合です.

Chicago は文字数は 4 以下ではありませんし最後が"n"ではありません. 条件 1，2 ともに不成立ですから偽です. Rome は文字数が 4 文字以下で，最後が"n"ではありませんから真です. London は文字数が 4 以下ではありませんが最後が"n"ですから真です. Bon は文字数は 4 以下ですが，最後が"n"ですから条件 1，2 がともに成立するため偽です.

② XOR 条件は次のようになります.

X	Y	X XOR Y
T	T	F
T	F	T
F	T	T
F	F	F

T はTRUE
F はFALSE

【プログラム例】

```
FOR i = 1 TO 8
    READ x$
    IF LEN(x$) <= 4 XOR RIGHT$(x$, 1) = "n" THEN PRINT x$
NEXT i
END
DATA Chicago,Paris,Rome,London
DATA Boston,Tokyo,Pittsburgh,Bon
```

（IF 行の注）文字数が 4 以下で最後が"n"以外かまたは文字数が 4 をこえ最後が"n"のとき表示

【結　果】

```
Rome
London
Boston
```

【例題 48】 EQV 条件

次のデータを使い，文字数が 4 以下で最後に"n"を持つものか，文字数が 4 以下ではなく，最後に"n"を持たないものを表示するプログラムをつくれ.

データ；Chicago,Paris,Rome,London,Boston,Tokyo,Pittsburgh,Bon

【解　説】 ◎ EQV 条件で分岐します.

① EQV は条件 1，2 がともに成立するかともに不成立のとき真です. したがって，本例では文字数が 4 以下で最後に"n"を持つか，文字数が 4 をこえ最後に"n"を持たないとき真です. 両方成立する文字列は Bon，両方不成立は Chicago，Paris，Tokyo，Pittsburgh です. この両ケースが真です. 片側が成立し，片側が不成立は Rome，London，Boston です. これは偽となります.

② EQV 条件は次のようになります.

X	Y	X EQV Y
T	T	T
T	F	F
F	T	F
F	F	T

TはTRUE
FはFALSE

【プログラム例】

```
FOR i = 1 TO 8
    READ x$
    IF LEN(x$) <= 4 EQV RIGHT$(x$, 1) = "n" THEN PRINT x$ ── 文字数が4以下で最後が"n"
                                                              かまたは文字数が4をこえ最
                                                              後が"n"ではないとき表示
NEXT i
END
DATA Chicago,Paris,Rome,London
DATA Boston,Tokyo,Pittsburgh,Bon
```

【結　果】

```
Chicago
Paris
Tokyo
Pittsburgh
Bon
```

> 【例題 49】　**IMP 条件**
>
> 次のデータを使い，文字数が4以下で最後に"n"を持たないもの以外を表示するプログラムをつくれ．
>
> データ；Chicago,Paris,Rome,London,Boston,Tokyo,Pittsburgh,Bon

【解　説】　◎　IMP 条件で分岐します．

① IMP は条件1が真で条件2が偽のときのみ偽で，他はすべて真です．本例では Rome だけが偽です．

② IMP 条件は次のようになります．

X	Y	X IMP Y
T	T	T
T	F	F
F	T	T
F	F	T

TはTRUE
FはFALSE

【プログラム例】

```
FOR i = 1 TO 8
    READ x$
```

```
    IF LEN(x$) <= 4 IMP RIGHT$(x$, 1) = "n" THEN PRINT x$ ―― 文字数が4以下で最後が"n"
NEXT i                                                      でないもの以外を表示
END
DATA Chicago, Paris, Rome, London
DATA Boston, Tokyo, Pittsburgh, Bon
```

【結　果】

```
Chicago
Paris
London
Boston
Tokyo
Pittsburgh
Bon
```

【例題50】 論理演算子の組合わせ

次のデータを使い，文字数が5以下で最後が"n"のものか，最後が"o"のものを表示するプログラムをつくれ.

データ; Chicago,Paris,Rome,London,Boston,Tokyo,Pittsburgh,Bon

【解　説】 ◎ 論理演算子を組合わせて条件判断し分岐します.

① 第1の条件は文字数が5以下でかつ最後が"n"のものです. この条件にあうものはBonのみです.

② 第2の条件は最後が"o"です. この条件にあうものはChicago, Tokyoです.

③ 条件1と条件2のORはBon, Chicago, Tokyoとなります.

【プログラム例】

```
FOR i = 1 TO 8
    READ x$
    L = LEN(x$)
    p$ = RIGHT$(x$, 1)
    IF (L <= 5 AND p$ = "n") OR (p$ = "o") THEN PRINT x$ ―― 文字数が5以下で最後が"n"
NEXT i                                                     かまたは最後が"o"のものを
END                                                        表示
DATA Chicago, Paris, Rome, London
DATA Boston, Tokyo, Pittsburgh, Bon
```

【結　果】

```
Chicago
Tokyo
Bon
```

〔演習問題〕

(43) 下のデータを読み"P"で始まる都市名を表示するプログラムをつくれ.

データ;Chicago, Paris, Rome, London, Boston, Tokyo, Pittsburgh, Bon

(44) (43)のデータを使い最後が"n"で終わる都市名を表示するプログラムをつくれ.

(45)　(43) のデータを使い5文字でかつ"P"で始まる都市名を表示するプログラムをつくれ.

(46)　(43) のデータを使い6文字以上のとき都市名と文字数を，それ以外では都市名のみを表示するプログラムをつくれ.

(47)　アルファベットA～Z，a～zを入力し，Hとそれより後の文字のときはそのまま表示し，それより前のときは小文字にして表示するプログラムをつくれ.

(48)　xを入力して正のときyを入力しx*yを表示し，xが負のときはx^3を表示するプログラムをつくれ.

(49)　xを入力して，xが正のときaを2，bを5，cを3とし，xが負のときaを4，bを2，cを7として$y=ax^2+bx+c$を求めて表示するプログラムをつくれ.

(50)　(43) のデータを使い文字数が最も少ない都市名を表示するプログラムをつくれ.

(51)　(43) のデータを使い"B"または"P"で始まる都市名を表示するプログラムをつくれ.

(52)　文字列を入力し，下のデータの中にあれば表示し，なければ"END"と表示するプログラムをつくれ.

データ;Tokyo, Osaka, Nagoya, Fukuoka, Sapporo, Okayama

(53)　(43) のデータを使って，文字数が4未満か8をこえるものを表示するプログラムをつくれ.

(54)　(43) のデータを使って，文字数が4以下か8をこえるものでかつ最後が"n"のものを表示するプログラムをつくれ.

(55)　(43) のデータを使って，文字数が6をこえて最後が"n"のものか，文字数が6をこえ，最後が"n"でないものを表示するプログラムをつくれ.

(56)　(43) のデータを使って，文字数が6をこえてChicagoではないものを表示するプログラムをつくれ.

〔解　答〕

(43)【プログラム例】

```
FOR i = 1 TO 8 ───────────────── データの読み込みのくり返し
    READ x$ ──────────────────── x$にデータを読み込む
    IF ASC(x$) = 80 THEN PRINT x$ ─┐ "P"で始まるときx$を表示
NEXT i ────────────────────────┘
END
DATA Chicago,Paris,Rome,London
DATA Boston,Tokyo,Pittsburgh,Bon
```

【結　果】

```
Paris
Pittsburgh
```

(44)　【プログラム例】

```
FOR i = 1 TO 8 ──────────────────────── データの読み込みのくり返し
    READ x$ ─────────────────────────── x$にデータを読み込む
    IF RIGHT$(x$, 1) = "n" THEN PRINT x$ ─┬ 右端がnのとき表示
NEXT i ───────────────────────────────┘
END
DATA Chicago, Paris, Rome, London
DATA Boston, Tokyo, Pittsburgh, Bon
```

【結　果】

```
London
Boston
Bon
```

(45)　【プログラム例】

```
FOR i = 1 TO 8 ──────────────────────── データの読み込みのくり返し
    READ x$ ─────────────────────────── x$にデータを読み込む
    l = LEN(x$) ─────────────────────── x$の長さをlに代入
    IF l = 5 AND ASC(x$) = 80 THEN PRINT x$ ─┬ lが5で先頭文字が"P"のとき表示
NEXT i ──────────────────────────────────┘
END
DATA Chicago, Paris, Rome, London
DATA Boston, Tokyo, Pittsburgh, Bon
```

【結　果】

```
Paris
```

(46)　【プログラム例】

```
FOR i = 1 TO 8 ─────────────────────────────────────── 8回のくり返し
    READ x$ ────────────────────────────────────────── x$にデータを読む
    IF LEN(x$) >= 6 THEN PRINT x$; LEN(x$) ELSE PRINT x$ ── 6文字以上のとき都市名と文字数を
NEXT i ─────────────────────────────────────────────── 表示，それ以外は都市名のみ表示
END
DATA Chicago, Paris, Rome, London
DATA Boston, Tokyo, Pittsburgh, Bon
```

【結　果】

```
Chicago 7
Paris
Rome
London 6
Boston 6
Tokyo
Pittsburgh 10
Bon
```

(47)　【プログラム例】

```
INPUT x$ ─────────────────────────────────────────── x$の入力
IF x$ >= "H" THEN PRINT x$ ELSE PRINT CHR$(ASC(x$) + 32) ── "H"とそれより後のとき表示，それ
END                                                          以外は小文字にして表示
```

【結　果】

```
? A                     Aを入力の例
a

? P                     Pを入力の例
P

? d                     dを入力の例
d
```

(48)【プログラム例】

```
INPUT x ──────────────── xの入力
IF x >= 0 THEN ───────── xが正のとき
    INPUT y ──────────── yを入力して，x*yを表示
    PRINT x * y
ELSE ─────────────────── xが負のとき
    PRINT x * x * x ──── x³を表示
END IF
END
```

【結　果】

```
? 6                      xに6，y=7を入力の例
? 7
 42

? -88                    xに-88を入力の例
-681472
```

(49)【プログラム例】

```
INPUT x ──────────────────────────────── xの入力
IF x >= 0 THEN ───────────────────────── xが正のとき
    a = 2: b = 5: c = 3 ──────────────── a=2，b=5，c=3とする
ELSE ─────────────────────────────────── xが負のとき
    a = 4: b = 2: c = 7 ──────────────── a=4，b=2，c=7とする
END IF
y = a * x * x + b * x + c ────────────── y=ax²+bx+cを求める
PRINT a; "x^2+"; b; "x+"; c; "="; y ──── 結果の表示
END
```

【結　果】

```
? 8                              8を入力の例
 2 x^2+ 5 x+ 3  =  171

? -3                             -3を入力の例
 4 x^2+ 2 x+ 7  =  37
```

(50)【プログラム例】

```
min = 20 ───────────── 最小の文字の都市名を求める
FOR i = 1 TO 8
    READ x$
    l = LEN(x$)
    IF l <= min THEN ──── 文字数がそのときの最小値 m より小さければ
        min = l              その文字数を m に代入，そのときの都市名を
        y$ = x$              y$に代入
END IF
NEXT i
PRINT y$
END
DATA Chicago,Paris,Rome,London
DATA Boston,Tokyo,Pittsburgh,Bon
```

【結　果】

```
Bon
```

(51)【プログラム例】

```
FOR i = 1 TO 8 ─────────────── データの読み込みのくり返し
    READ x$ ────────────────── x$にデータを読み込む
    IF ASC(x$) = 66 THEN ───── "B"で始まるとき表示
        PRINT x$
    ELSEIF ASC(x$) = 80 THEN ─ "P"で始まるとき表示
        PRINT x$
END IF
NEXT i
END
DATA Chicago,Paris,Rome,London
DATA Boston,Tokyo,Pittsburgh,Bon
```

【結　果】

```
Paris
Boston
Pittsburgh
Bon
```

(52)【プログラム例】

```
INPUT x$ ──────────────────────── x$の入力
DO UNTIL x$ = y$ ──────────────── x$=y$で終了，同じデータがあればループの外へ出る
    READ y$ ───────────────────── データを読む
    IF y$ = "End" THEN x$ = y$ ── データを読んで"END"になれば x$を"END"にしてル
LOOP                               ープの外へ出る
PRINT x$ ──────────────────────── x$の表示
END
DATA Tokyo,Osaka,Nagoya
DATA Fukuoka,Sapporo,Okayama
DATA End
```

【結　果】

```
? Sapporo              Sapporoを入力の例
Sapporo

? Kyoto                Kyotoを入力の例
End
```

(53)　【プログラム例】

```
FOR i = 1 TO 8
    READ x$
    l = LEN(x$)
    IF (l < 4) OR (l > 8) THEN PRINT x$――文字数が4未満か8をこえるものを表示
NEXT i
END
DATA Chicago, Paris, Rome, London
DATA Boston, Tokyo, Pittsburgh, Bon
```

【結　果】

```
Pittsburgh
Bon
```

(54)　【プログラム例】

```
FOR i = 1 TO 8
    READ x$
    l = LEN(x$)
    p$ = RIGHT$(x$, 1)
    IF (l <= 4 OR l > 8) AND (p$ = "n") THEN PRINT x$――文字数が4以下か8をこえ,
NEXT i                                                かつ最後が"n"のとき表示
END
DATA Chicago, Paris, Rome, London
DATA Boston, Tokyo, Pittsburgh, Bon
```

【結　果】

```
Bon
```

(55)　【プログラム例】

```
FOR i = 1 TO 8
    READ x$
    IF LEN(x$) > 6 EQV RIGHT$(x$, 1) = "n" THEN PRINT x$――文字数が6をこえ最後が"n"
NEXT i                                                 または文字数が6以下で最後
END                                                    が"n"でないとき表示
DATA Chicago, Paris, Rome, London
DATA Boston, Tokyo, Pittsburgh, Bon
```

【結　果】

```
Paris
Rome
Tokyo
```

(56) **【プログラム例】**

```
FOR i = 1 TO 8
    READ x$
    IF LEN(x$) > 6 AND NOT (x$ = "Chicago") THEN PRINT x$ ─── 文字数が6をこえかつ
NEXT i                                                         Chicagoでないとき
END                                                            表示
DATA Chicago,Paris,Rome,London
DATA Boston,Tokyo,Pittsburgh,Bon
```

【結　果】

```
Pittsburgh
```

7章　多分岐

SELECT～CASE, END SELECT

【例題 51】　SELECT～CASE による多分岐

数値を入力し，1のとき"A"，2のとき"B"，3のとき"C"，4のとき"D"，5のとき"E"を表示し，それ以外では"END"と表示するプログラムをつくれ.

【解　説】　◎　SELECT～CASE による多分岐をします.

① 条件によって分岐先を3つ以上にしたい場合 SELECT～CASE を使います.

② 本例は数値を入力し，1，2，3，4，5の場合とそれ以外の場合の6つの分岐をします．その形は次のとおりです.

```
SELECT x%            ―――――  x%の値によって分岐先が決まる
   CASE  1       ――┐  x%が1のとき
        PRINT"A" ――┘  x%が1のときの処理．複数の文を書いてよい．終了す
                        るとEND SELECTの次へ進む
   CASE  2       ――┐  x%が2のとき
        PRINT"B" ――┘
   CASE  3       ――┐  x%が3のとき
        PRINT"C" ――┘
   CASE  4       ――┐  x%が4のとき
        PRINT"D" ――┘
   CASE  5       ――┐  x%が5のとき
        PRINT"E" ――┘
   CASE ELSE     ――┐  1，2，3，4，5以外のとき
        PRINT"END"―┘
END SELECT           ―――――  SELECT文の終了
```

③ x%が1のときはCASE1へ進み"A"を表示してEND SELECTの次へ進みます．PRINT"A"の部分が1の場合の処理内容です．ここは複数の文をかくことができます．x%が2，3，4，5の場合は各々CASE2，CASE3，CASE4，CASE5へ分岐し，各々の処理をします.

④ x%が1，2，3，4，5以外ではCASE ELSEへ分岐し，その処理をします．x%が1，2，3，4，5しかなければCASE ELSEはいりませんが，x%が1～5以外があって，CASE ELSEがないとエラーとなります.

⑤　従来のBASICではON～GOSUBまたはON～GOTOでかくことができますが，かなりみにくいものとなります.

ON～GOSUBを使った例

(◯◯などは行番号を示す)

```
    ON x%  GOSUB  ◯◯, ××, △△, □□, ●●, ▲▲
       ≀
◯◯  PRINT"A"
    RETURN
××  PRINT"B"
    RETURN
△△  PRINT"C"
    RETURN
□□  PRINT"D"
    RETURN
●●  PRINT"E"
    RETURN
▲▲  PRINT"END"
    RETURN
```

ON～GOTOを使った例

(◯◯などは行番号を示す)

```
    ON x%  GOTO  ◯◯, ××, △△, □□, ●●, ▲▲
■■     ≀
◯◯  PRINT"A"
    GOTO ■■
××  PRINT"B"
    GOTO ■■
△△  PRINT"C"
    GOTO ■■
□□  PRINT"D"
    GOTO ■■
●●  PRINT"E"
    GOTO ■■
▲▲  PRINT"END"
    GOTO ■■
```

【プログラム例】

```
INPUT x% ──────────────── 整数の入力
SELECT CASE x% ─────────── x%の値によって分岐
    CASE 1──────────────┐ x%が1のとき
        PRINT "A"───────┘
    CASE 2 ─────────────┐ x%が2のとき
        PRINT "B"───────┘
    CASE 3 ─────────────┐ x%が3のとき
        PRINT "C"───────┘
    CASE 4 ─────────────┐ x%が4のとき
        PRINT "D"───────┘
    CASE 5 ─────────────┐ x%が5のとき
        PRINT "E"───────┘
    CASE ELSE ──────────┐ x%が1～5以外のとき
        PRINT "END"─────┘
END SELECT ─────────────── 分岐の終了
END
```

【結　果】

```
? 3            3を入力の例
C

? 5            5を入力の例
E

? 15           15を入力の例
END
```

【例題52】　文字による多分岐をします

文字を入力し，"A"のとき"Apple"，"B"のとき"Bench"，"C"のとき"Chair"，"D"のとき"Desk"，"Eのとき"Election"，それ以外では"END"と表示するプログラムをつくれ．

【解　説】　◎　SELECT文字CASEによって多分岐をします．

① 文字を条件として多分岐をします．この場合はCASE "A"のように文字を""でかこみます．あとは数値の場合と同じです．

【プログラム例】

```
INPUT x$ ──────────────── x$の入力
SELECT CASE x$ ─────────── x$の値によって分岐
    CASE "A"────────────┐ x$が"A"のとき
        PRINT "Apple"───┘
    CASE "B"────────────┐ x$が"B"のとき
        PRINT "Bench" ──┘
    CASE "C"────────────┐ x$が"C"のとき
        PRINT "Chair"───┘
    CASE "D"────────────┐ x$が"D"のとき
        PRINT "Desk"────┘
```

```
    CASE "E"──────────────┐ x$が"E"のとき
        PRINT "Election"──┘
    CASE ELSE ────────────┐ x$がA～E以外のとき
        PRINT "END" ──────┘
END SELECT ──────────────── 分岐の終了
END
```

【結　果】

```
? B              Bを入力の例
Bench

? a              aを入力の例
END
```

【例題 53】　範囲で分岐

数値を入力し，1000 以上のとき"x>=1000"，500 から 999 までのとき"500<=x<1000"，300 から 499 のとき"300<=x<500"，100 から 299 のとき"100<=x<300"，0 から 99 までのとき"0<=x<100"，それ以外では"マイナス"と表示するプログラムをつくれ．ただし，上の表示のうち x は各々実際に入力された値を表示するものとする．たとえば，125 が入力されれば"100<=125<300"と表示するものとする．

【解　説】　◎　CASE の後に範囲を指定します．

① CASE IS　>=1000 では値が 1000 以上の場合の分岐先です．IS　>=1000 が 1000 以上を示します．比較演算子として>，>=，<，<=，<>，=が使えます．ただし，=はたとえば，CASE　5 のように単に値をかくだけですみます．

② CASE　500　TO　999 は 500 から 999 のときの分岐先です．値 1　TO　値 2 で値 1 から値 2 までの範囲を指定できます．ただし，必ず値 1＜値 2 である必要があります．

【プログラム例】

```
INPUT x% ──────────────────────────── x%の入力
SELECT CASE x% ─────────────────────── x%の値によって分岐
    CASE IS >= 1000 ─────────────────┐ x%が1000以上のとき
        PRINT x%; ">=1000"───────────┘
    CASE 500 TO 999 ─────────────────┐ x%が500～999のとき
        PRINT "500<="; x%; "<1000" ──┘
    CASE 300 TO 499 ─────────────────┐ x%が300～499のとき
        PRINT "300<="; x%; "<500"────┘
    CASE 100 TO 299 ─────────────────┐ x%が100～299のとき
        PRINT "100<="; x%; "<300"────┘
    CASE 0 TO 99 ────────────────────┐ x%が0～99のとき
        PRINT "0<="; x%; "<100" ─────┘
    CASE ELSE ─────────────────────── x%が0未満のとき
        PRINT "ﾏｲﾅｽ"
END SELECT──────────────────────────── 分岐の終了
END
```

【結　果】

```
? 1563                    1563を入力の例
 1563 >=1000

? 48                      48を入力の例
0<= 48 <100

? -1200                   -1200を入力の例
マイナス
```

【例題54】　CASEに複数の範囲を指定します

整数を入力し，負のときはその値を，1から10，20から30，40から50のときは2乗の値，11から19，21から29，31から39のときは平方根，50をこえるときはその値を表示するプログラムをつくれ．

【解　説】　◎　CASEに複数個の範囲を指定します．

① CASE 1 TO 10, 20 TO 30, 40 TO 50によって1から10，20から30，40から50の3つの範囲を指定することができます．

【プログラム例】

```
INPUT x% ―――――――――――――――――――――――― x%の入力
SELECT CASE x% ――――――――――――――――――― x%の値によって分岐
    CASE IS <= 0 ――――――――――――――――― x%が0以下のとき
        PRINT x%
    CASE 1 TO 10, 20 TO 30, 40 TO 50 ―― x%が1～10，20～30，40～50のとき
        PRINT x% * x%
    CASE 11 TO 19, 21 TO 29, 31 TO 39 ― x%が11～19，21～29，31～39のとき
        PRINT SQR(x%)
    CASE IS > 50 ――――――――――――――――― x%が50より大きいとき
        PRINT x%
END SELECT ――――――――――――――――――――――― 分岐の終了
END
```

【結　果】

```
? 25                      25を入力の例
 625

? 15                      15を入力の例
 3.872983

? -85                     -85を入力の例
-85
```

【例題 55】　多分岐でサブルーチンを呼び出す

次のメニュー画面を表示し，1のとき和，2のとき差，3のとき積，4のとき商，それ以外では再入力するプログラムをつくれ．ただし，数値が入力されたら演算サブルーチンへ各々分岐して，数を2つ入力して演算し，結果を表示するものとする．

```
和　＝1
差　＝2
積　＝3
商　＝4

1から4
```

【解　説】　◎　多分岐とサブルーチンでメインルーチンをすっきりさせます．

① メインルーチンは入力された値1～4とそれ以外で各々多分岐するものです．【例題 51】と同じ構造です．

② 各々の case では call a1(a,b,c)～a4(a,b,c)でサブルーチン a1～a4 を呼び出します．

③ 結果はメインルーチンの最後で表示します．

【プログラム例】

```
DECLARE SUB a1 (x!, y!, z!)
DECLARE SUB a2 (x!, y!, z!)
DECLARE SUB a3 (x!, y!, z!)
DECLARE SUB a4 (x!, y!, z!)
PRINT "和　=1"          ─┐ メニュー画面
PRINT "差　=2"           │
PRINT "積　=3"           │
PRINT "商　=4"           │
PRINT                   ─┘
re: INPUT "1から4"; x%  ── x%の入力
SELECT CASE x%          ── x%の値で分岐
    CASE 1              ─┐ 1のとき
        d$ = "+"         │
        CALL a1(a, b, c) ─┘
    CASE 2              ─┐ 2のとき
        d$ = "-"         │
        CALL a2(a, b, c) ─┘
    CASE 3              ─┐ 3のとき
        d$ = "*"         │
        CALL a3(a, b, c) ─┘
    CASE 4              ─┐ 4のとき
        d$ = "/"         │
        CALL a4(a, b, c) ─┘
    CASE ELSE           ─┐ 1～4以外のとき再入力
        GOTO re:        ─┘
END SELECT
PRINT a; d$; b; "="; c  ── 結果の表示
END
```

```
SUB a1 (x, y, z) ─────────┐─サブルーチン a1
    INPUT x, y            │
    z = x + y             │
END SUB ──────────────────┘

SUB a2 (x, y, z) ─────────┐─サブルーチン a2
    INPUT x, y            │
    z = x - y             │
END SUB ──────────────────┘

SUB a3 (x, y, z) ─────────┐─サブルーチン a3
    INPUT x, y            │
    z = x * y             │
END SUB ──────────────────┘

SUB a4 (x, y, z) ─────────┐─サブルーチン a4
    INPUT x, y            │
    z = x / y             │
END SUB ──────────────────┘
```

【結　果】

```
和   = 1                 3 を入力し，5，4 を入力した例
差   = 2
積   = 3
商   = 4

 1 から 4 ? 3
? 5,4
 5 * 4 = 20

和   = 1                 2 を入力し，7，3 を入力した例
差   = 2
積   = 3
商   = 4

 1 から 4 ? 2
? 7,3
 7 - 3 = 4

和   = 1                 4 を入力し，99，3 を入力した例
差   = 2
積   = 3
商   = 4

 1 から 4 ? 4
? 99,3
 99 / 3 = 33


和   = 1                 8,7,1 を入力し，50,24 を入力した例
差   = 2
積   = 3
商   = 4

 1 から 4 ? 8
 1 から 4 ? 7
 1 から 4 ? 1
? 50,24
 50 + 24 = 74
```

〔演習問題〕

(57) 数を入力し，1のとき"first"，2のとき"second"，3のとき"third"と表示し，それ以外では入力された値を表示するプログラムをつくれ.

(58) 文字を入力し，"A"のとき"ABC"，"B"のとき"BCD"，"C"のとき"CDE"，それ以外のとき"XYZ"と表示するプログラムをつくれ.

(59) 整数を入力し，65から90のときはその値をアスキーコードとする文字を，97から122までのときはその値をアスキーコードとする文字を表示し，それ以外では"アルファベット　イガイ"と表示するプログラムをつくれ.

(60) 整数を入力して1のとき，サブルーチン aa を呼び出し"ニュウリョクチ=1"，2のとき，サブルーチン bb を呼び出し"ニュウリョクチ=2"，それ以外のときはサブルーチン cc と呼び出し"リュウリョクチ　ハ　1,2　イガイ"と表示するプログラムをつくれ.

(61) 整数 x%を入力し，1～5のとき x%!（階乗)，6～9のとき 2x%（累乗)，10～15のとき，1からx%までの和を求めそれ以外は再入力するプログラムをつくれ.

(62) 英文字列を入力し，"January"のときは"January　ハ　31　ニチ"，"Feburary"のとき"Ferbruary　ハ　28　ニチ"，…"June"のとき"June　ハ　30　ニチ"と表示し，その他のときは再入力するプログラムをつくれ.

(63) アルファベットの大文字または小文字，数字，またはカナを1つ入力し，小文字のとき"アルファベット　コモジ"，大文字のとき"アルファベット　オオモジ"，数字のとき"スウジ"，カナのとき"カナ"と表示するプログラムをつくれ.

〔解　答〕

(57) **【プログラム例】**

```
INPUT x%  ──────────── x%の入力
SELECT CASE x%
    CASE 1 ────────── 1のとき
        PRINT " first"
    CASE 2 ────────── 2のとき
        PRINT "second"
    CASE 3 ────────── 3のとき
        PRINT "third"
    CASE ELSE ─────── 1～3以外のとき
        PRINT x%
END SELECT
END
```

【結　果】

```
? 1          1を入力の例
 first

? 3          3を入力の例
third

? 120        120を入力の例
 120
```

(58) 【プログラム例】

```
INPUT x$ ──────────────── x$を入力
SELECT CASE x$
    CASE "A" ────────────┬─ Aのとき
        PRINT "ABC" ─────┘
    CASE "B" ────────────┬─ Bのとき
        PRINT "BCD" ─────┘
    CASE "C" ────────────┬─ Cのとき
        PRINT "CDE" ─────┘
    CASE ELSE ───────────┬─ A～C以外のとき
        PRINT "XYZ" ─────┘
END SELECT
END
```

【結　果】

```
? A            Aを入力の例
ABC

? f            fを入力の例
XYZ
```

(59) 【プログラム例】

```
INPUT x% ─────────────────────── x%の入力
SELECT CASE x%
    CASE 65 TO 90 ──────────────┬─ 65～90のとき
        PRINT CHR$(x%) ─────────┘
    CASE 97 TO 122 ─────────────┬─ 97～122のとき
        PRINT CHR$(x%) ─────────┘
    CASE ELSE ──────────────────┬─ その他のとき
        PRINT "ｱﾙﾌｧﾍﾞｯﾄ ｲｶﾞｲ" ───┘
END SELECT
END
```

【結　果】

```
? 69                69を入力の例
E

? 115               115を入力の例
s

? 500               500を入力の例
ｱﾙﾌｧﾍﾞｯﾄ ｲｶﾞｲ
```

(60) 【プログラム例】

```
DECLARE SUB aa () ──┬─ サブルーチンの宣言
DECLARE SUB bb () ──┤
DECLARE SUB cc () ──┘
INPUT x% ─────────── x%の入力
SELECT CASE x%
    CASE 1 ──────────┬─ 1のとき
        CALL aa ─────┘
    CASE 2 ──────────┬─ 2のとき
        CALL bb ─────┘
    CASE ELSE ───────┬─ 1～2以外のとき
        CALL cc ─────┘
END SELECT
```

```
END

SUB aa ──────────────── サブルーチン aa
PRINT "ﾆｭｳﾘｮｸﾁ=1"
END SUB

SUB bb ──────────────── サブルーチン bb
PRINT "ﾆｭｳﾘｮｸﾁ=2"
END SUB

SUB cc ──────────────── サブルーチン cc
PRINT "ﾆｭｳﾘｮｸﾁ ﾊ 1,2 ｲｶﾞｲ"
END SUB
```

【結　果】

```
? 2                          2を入力の例
ﾆｭｳﾘｮｸﾁ=2

? 1                          1を入力の例
ﾆｭｳﾘｮｸﾁ=1

? 453                        453を入力の例
ﾆｭｳﾘｮｸﾁ ﾊ 1,2 ｲｶﾞｲ
```

(61)【プログラム例】

```
re: INPUT x% ─────────────── x%の入力
SELECT CASE x% ───────────── x%の値で分岐
    CASE 1 TO 5 ──────────── 1～5のとき
        z% = 1                z%に階乗を求める
        FOR i% = x% TO 1 STEP -1
            z% = z% * i%
        NEXT i%
        PRINT z%
    CASE 6 TO 9 ──────────── 6から9のとき
        PRINT 2 ^ x%          2^x% を表示
    CASE 10 TO 15 ────────── 10から15のとき
        w% = 0                wa%に和を求める
        FOR i% = 1 TO x%
            wa% = wa% + i%
        NEXT i%
        PRINT wa%
    CASE ELSE ────────────── 1～15以外のとき再入力
        GOTO re:
END SELECT
END
```

【結　果】

```
? 9             9を入力の例
 512
? 4             4を入力の例
 24
? 14            14を入力の例
 105
? 85            85を入力し，再入力で1を入力の例
? 1
 1
```

(62) **【プログラム例】**

```
re: INPUT x$ ─────────────────── 文字列の入力
SELECT CASE x$ ───────────────── 文字列x$によって分岐
    CASE "January" ──────────┐── "January"のとき
        PRINT "January ハ 31 ニチ"┘
    CASE "February" ─────────┐── "February"のとき
        PRINT "February ハ 28 ニチ"┘
    CASE "March" ────────────┐── "March"のとき
        PRINT "March ハ 31 ニチ"─┘
    CASE "April" ────────────┐── "April"のとき
        PRINT "April ハ 30 ニチ"─┘
    CASE "May" ──────────────┐── "May"のとき
        PRINT "May ハ 31 ニチ"───┘
    CASE "June" ─────────────┐── "June"のとき
        PRINT "June ハ 30 ニチ"──┘
    CASE ELSE ───────────────┐── その他のとき再入力
        GOTO re: ────────────┘
END SELECT
END
```

【結　果】

```
? May                    Mayを入力の例
May ハ 31 ニチ
? February               Februaryを入力の例
February ハ 28 ニチ
```

(63) **【プログラム例】**

```
re: INPUT x$ ─────────────────── 文字列x$の入力
xx% = ASC(x$) ────────────────── x$のアスキーコードをxx%に代入
SELECT CASE xx% ──────────────── xx%の値で分岐
    CASE 48 TO 57 ───────────┐── 48～57のとき（"0"～"9"のとき）
        PRINT "スウジ" ──────┘
    CASE 65 TO 90 ───────────┐── 65～90のとき（"A"～"Z"のとき）
        PRINT "アルファベット オオモジ"┘
    CASE 97 TO 122 ──────────┐── 97～122のとき（"a"～"z"のとき）
        PRINT "アルファベット コモジ"─┘
    CASE 166 TO 221 ─────────┐── 166～221のとき（"ア"～"ン"のとき）
        PRINT "カナ" ────────┘
```

```
    CASE ELSE ──────────────┐── その他のとき再入力
        GOTO re: ───────────┘
END SELECT
END
```

【結　果】

```
? 65 ──────────────── 65を入力の例
ｽｳｼﾞ
? A ───────────────── Aを入力の例
ｱﾙﾌｧﾍﾞｯﾄ ｵｵﾓｼﾞ
? x ───────────────── xを入力の例
ｱﾙﾌｧﾍﾞｯﾄ ｺﾓｼﾞ
? ｼ ───────────────── シを入力の例
ｶﾅ
```

8章 配　　列

DIM, OPTION BASE

【例題 56】　配　列

下のデータを配列 a%(0)〜a%(9)に代入して，a%(5)と a%(7)を表示するプログラムをつくれ.

データ;7, 5, 2, 9, 7, 3, 8, 4, 1, 0

【解　説】　◎　1次数値配列を使います.

① DIM a%(10)で a%(0), a%(1), …a%(9)の 10 個の整数型配列変数を確保します. これは DIM a(10)　AS INTEGER と宣言することもできます.

② FOR i=0　TO　9

　　READ a%(i)

　NEXT i

によってデータを順に a%(0), a%(1), …a%(9)へ読み込みます. 1度データが配列変数に代入されればあとはどれでも任意にとり出せます.

③ PRINT a%(5);a%(7)によって a%(5)の 3 と a%(7)の 4 を表示します.

④ a%(10)の 10 を添字といい最大 32,768 まで使えます. 整数型の場合 2 バイトを使用しますから a%(32,768)は約 65k バイト使用します.

【プログラム例】

```
DIM a%(10) ─────────── 配列 a%(0)〜a%(9)の宣言
FOR i = 0 TO 9 ──────┐ i を 0〜9 までループ
    READ a%(i) ──────┼ a%(0)〜a%(9)にデータを読み込む
NEXT i ──────────────┘
PRINT a%(5); a%(7) ──── a%(5)と a%(7)の表示
END
DATA 7,5,2,9,7,3,8,4,1,0
```

【結　果】

```
3  4
```

【例題 57】　添字を任意に決定

添字を−5 から 5 までとする配列 c%()を宣言して，下のデータを順に代入し，c%(0)の値を表示するプログラムをつくれ.

データ;3, 8, 7, 2, 1, 9, 6, 3, 5, 2, 7

【解　説】　◎　添字の範囲を任意に決めます.

① DIM c%(-5　TO　5)は c%(-5), c%(-4), c%(-3), …c%(0), c%(1), …c%(5)の 11 個の領域を確保します.

Quick BASIC では添字の下限と上限を自由に設定できるようになった他マイナスを使うこともできます. この便利さについては次の【例題 58】で示します.

【プログラム例】

```
DIM c%(-5 TO 5) ──────────── 配列 c%の宣言
FOR i = -5 TO 5 ──────────┐ c%(-5)～c%(5)にデータを読み込む
    READ c%(i)             │
NEXT i ────────────────────┘
PRINT c%(0); ─────────────── c%(0)の表示
END
DATA 3,8,7,2,1,9,6,3,5,2,7
```

【結　果】

```
9
```

> 【例題 58】　x を－3 から＋3 まで $y=x^3+x^2+x$ を求める
>
> x^3+x^2+x の値を x を-3 から+3 まで増分を 1 として求め配列 y%( )に格納して表示するプログラムをつくれ.

【解　説】　◎　添字にマイナス値を使って関数の値を求めます.

① x^3+x^2+x の値を x を-3 から+3 までについて求めます. このとき, 結果を入れる配列を y%( )とすると FOR x=-3　TO　3 のくり返しで x を-3, -2, -1, 0, 1, 2, 3 として, 関数の値を求め, それを y%(x)に代入できます. 添字として x をそのまま使えますから非常に簡単です.

② 添字を 0～6 として求める従来の BASIC では次のようになります.

```
FOR x=-3  TO  3
    Y(x+3)=x^3+x^2+x            ←  x+3 によって添字を 0, 1, 2, …6 としている.
NXET I
```

【プログラム例】

```
DIM y%(-3 TO 3) ─────────── 配列 y%(-3)～y%(3)の宣言
FOR x = -3 TO 3 ──────────┐ x を-3 から 3 まで x³+x²+x を求めて y%(-3)～y%(3)に代入
    y%(x) = x * x * x + x * x + x
NEXT x ────────────────────┘
FOR x = -3 TO 3 ──────────┐ y%(-3)～y%(3)の表示
    PRINT y%(x); ─────────┘
NEXT x
END
```

【結　果】

```
-21 -6 -1  0  3  14  39
```

【例題 59】　1次元文字列配列とオプションベース

文字列配列 b$(1)～b$(5)に下のデータを読み，最初と最後を表示するプログラムをつくれ.

データ;Yamada, Osaki, Ueda, Yoshino, Honda

【解　説】 ◎ 文字列配列とオプションベースを使います.

① OPTION BASE 1 によって，配列の添字を1からとります．したがって，DIM b$(5)は b$(1)，b$(2)…b$(5)の5個の領域を確保します.

② 配列の添字0を使わない方がわかりやすい場合があります．その場合は OPTION BASE 1 とするとメモリが無駄になりません.

【プログラム例】

```
OPTION BASE 1 ─────────────── 添字を1から始める
DIM b$(5) ─────────────────── 配列 b$(1)～b$(5)の宣言
FOR i = 1 TO 5 ────────────┐─ b$(1)～b$(5)にデータを読み込む
    READ b$(i)             │
NEXT i ────────────────────┘
PRINT b$(1); "  "; b$(5) ───── b$(1)と b$(5)の表示
END
DATA Yamada,Osaki,Ueda,Yoshino,Honda
```

【結　果】

```
Yamada  Honda
```

【例題 60】　2次元配列

下のデータ組を2次元配列に読み表示するプログラムをつくれ.

データ; JAPAN　　TOKYO
USA　　WASHIGTON
UK　　LONDON
FRANCE　　PARIS
ITALY　　ROME
GERMANY　　BON

【解　説】 ◎ 2次元配列を使います.

① DIM x$(6, 2)は2次元の文字列配列（6行2列）を次のようにとります．6*2=12 個の領域を確保します.

	0	1
0	x$ (0,0)	x$ (0,1)
1	x$ (1,0)	x$ (1,1)
2	x$ (2,0)	x$ (2,1)
3	x$ (3,0)	x$ (3,1)
4	x$ (4,0)	x$ (4,1)
5	x$ (5,0)	x$ (5,1)

②
```
FOR i=0  TO  5
    FOR j=0  TO  1
        READ x$(i,j)
    NEXT j
NEXT i
```

これが２次元配列へのデータの読み込みの普通の仕方です．

③　x$(6,2)は２次元ですが x$(5,3,8,9,4)のように５次元の配列も使えます．最大 60 次元まで使えます．３次元配列の例を【例題 62】，４次元の例を【例題 63】で示します．

【プログラム例】

```
DIM x$(6, 2) ──────────────── 2次元配列の宣言
FOR i = 0 TO 5 ─────────────── 配列 x$にデータを読み込む
    FOR j = 0 TO 1
        READ x$(i, j)
NEXT j, i
FOR i = 0 TO 5 ─────────────── 表示
    FOR j = 0 TO 1
        PRINT USING "&        &"; x$(i, j);
    NEXT j
PRINT
NEXT i
END
DATA JAPAN,TOKYO,USA,WASHINGTON
DATA UK,LONDON,FRANCE,PARIS
DATA ITALY,ROME,GERMANY,BON
```

【結　果】

```
JAPAN      TOKYO
USA        WASHINGTO
UK         LONDON
FRANCE     PARIS
ITALY      ROME
GERMANY    BON
```

【例題 61】　数値と文字の配列

下の表を配列に代入し，金額を求めて，金額を含む表を表示するプログラムをつくれ．

商　　品	数　　量	単価（千円）
パソコン	25	458
ワープロ	48	216
ＣＤ	21	142
ＶＴＲ	30	98
ビデオカメラ	13	126

【解　説】 ◎ 数値と文字列の配列を考えます.

① 数値と文字列の混合した配列をつくることはできません. 一度すべて文字として文字配列に代入し, 数値としたい数字の部分は文字を数値化して使います.

② 全体を文字列配列 s$(5,3)に代入します. 0列目は商品名, 1列目は数量, 2列目は単価です. ただし, 3列目に金額を代入するため宣言としては s$(5,4)とします.

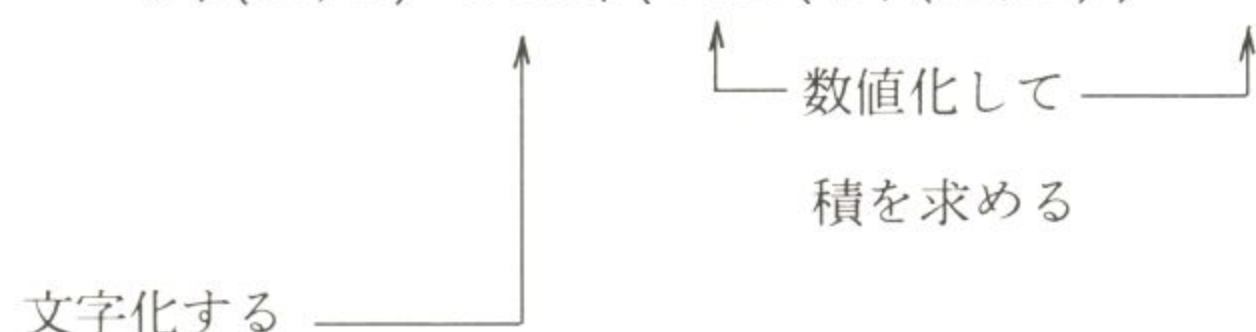
③ s$(0,0)〜s$(4,2)に文字としてデータをすべて読み込みます.

④ 金額の項は第1列と第2列の積です. したがって, 第1列と第2列を読み出して, それを数値化して積を求め, 次にそれを文字化して3列目に格納します.

たとえば, s$(2,3)は次のように求めます.

```
s$(2,3)=STR$(VAL(s$(2,1))*VAL(s$(2,2)))
```

数値化して積を求める（VAL(s$(2,1))〜VAL(s$(2,2))）

文字化する（STR$）

【プログラム例】

```
DIM s$(5, 4)                                      ― 配列の宣言
FOR i = 0 TO 4                                    ― 0〜4行についてデータ読み込みと代入
    FOR j = 0 TO 2                                ― 0〜2列を配列へ読み込む
        READ s$(i, j)
    NEXT j
    s$(i, 3) = STR$(VAL(s$(i, 1)) * VAL(s$(i, 2))) ― 3列の計算をして文字化して代入
NEXT i
PRINT "ｼｮｳﾋﾝ          ｽｳﾘｮｳ     ﾀﾝｶ     ｷﾝｶﾞｸ"     ― 結果の表示
FOR i = 0 TO 4
    PRINT USING "&        & "; s$(i, 0);
    FOR j = 1 TO 3
        PRINT USING "##,###     "; VAL(s$(i, j));
    NEXT j
PRINT
NEXT i
END
DATA ﾊﾟｿｺﾝ,25,458,ﾜｰﾌﾟﾛ,48,216,CD,21,142
DATA VTR,30,98,ﾋﾞﾃﾞｵｶﾒﾗ,13,126
```

【結　果】

```
ｼｮｳﾋﾝ          ｽｳﾘｮｳ     ﾀﾝｶ     ｷﾝｶﾞｸ
ﾊﾟｿｺﾝ          25        458     11,450
ﾜｰﾌﾟﾛ          48        216     10,368
CD             21        142      2,982
VTR            30         98      2,940
ﾋﾞﾃﾞｵｶﾒﾗ       13        126      1,638
```

【例題 62】　3 次元配列

次の表を 3 次元配列に代入し，次に，地区名を入力して，その地区のデータを表示するプログラムをつくれ．

地　区	商　品	数　量	金　額	利　益	1 人当たり売　上
東　京	パソコン	350	500	170	100
	ワープロ	650	800	120	180
大　阪	パソコン	250	300	110	120
	ワープロ	400	500	100	110
名古屋	パソコン	210	230	104	115
	ワープロ	350	450	105	109

【解　説】　◎　3 次元配列をつくります．

① 3 次元配列は"地区"の項，"商品"の項，"数量，金額，利益，1 人当たり売上"の項の 3 項目です．第 1 の要素は"東京"，"大阪"，"名古屋"からなります．第 2 の要素は"パソコン"，"ワープロ"，からなります．第 3 の要素は"数量，金額，利益，1 人当たり売上"からなります．

② 配列の各要素は次のようにとられます．

地　区	商　品	数　量	金　額	利　益	1 人当たり売　上
東　京	パソコン	r%（0,0,0）	r%（0,0,1）	r%（0,0,2）	r%（0,0,3）
	ワープロ	r%（0,1,0）	r%（0,1,1）	r%（0,1,2）	r%（0,1,3）
大　阪	パソコン	r%（1,0,0）	r%（1,0,1）	r%（1,0,2）	r%（1,0,3）
	ワープロ	r%（1,1,0）	r%（1,1,1）	r%（1,1,2）	r%（1,1,3）
名古屋	パソコン	r%（2,0,0）	r%（2,0,1）	r%（2,0,2）	r%（2,0,3）
	ワープロ	r%（2,1,0）	r%（2,1,1）	r%（2,1,2）	r%（2,1,3）

配列名を r%とする

r%(x,y,z)

- z：数量が 0，金額が 1，利益が 2，1 人当たり売上が 3
- y：パソコンが 0，ワープーロが 1
- x：東京が 0，大阪が 1，名古屋が 2

③ 地区別が入力されたら，東京の場合は c%を 0 として，大阪の場合は c%を 1 として，名古屋の場合は c%を 2 として r%(c%,0,0)から r%(c%,1,3)までを表示します．

【プログラム例】

```
DIM r%(3, 2, 4)────────────────3 次元配列の宣言
FOR i = 0 TO 2─────────────────データを配列へ読み込む
    FOR j = 0 TO 1
        FOR k = 0 TO 3
            READ r%(i, j, k)
        NEXT k
    NEXT j
NEXT i
```

```
INPUT "ﾁｸ ", t$ ――――――――――――――――――――――地区名の入力
IF t$ = "ﾄｳｷｮｳ" THEN c% = 0 ―――――――― t$の内容によって c%の値を 0～2 とする
IF t$ = "ｵｵｻｶ" THEN c% = 1
IF t$ = "ﾅｺﾞﾔ" THEN c% = 2
PRINT t$ ――――――――――――――――――――――――――――――――――――結果の表示
PRINT "        ｽｳﾘｮｳ   ｷﾝｶﾞｸ   ﾘｴｷ   ｳﾘｱｹﾞ/ﾆﾝ"
PRINT "ﾊﾟｿｺﾝ ";
PRINT r%(c%, 0, 0); "   "; r%(c%, 0, 1); "   "; r%(c%, 0, 2); r%(c%, 0, 3)
PRINT "ﾜｰﾌﾟﾛ ";
PRINT r%(c%, 1, 0); "   "; r%(c%, 1, 1); "   "; r%(c%, 1, 2); r%(c%, 1, 3)
END
DATA 350,500,170,100,650,800,120,180
DATA 250,300,110,120,400,500,100,110
DATA 210,230,104,115,350,450,105,109
```

【結　果】

```
チク ナゴヤ                                     ナゴヤを入力の例
ナゴヤ
        スウリョウ  キンガク  リエキ  ウリアゲ/ニン
パソコン  210      230      104    115
ワープロ  350      450      105    109
チク トウキョウ
トウキョウ                                      トウキョウを入力の例
        スウリョウ  キンガク  リエキ  ウリアゲ/ニン
パソコン  350      500      170    100
ワープロ  650      800      120    180
```

【例題 63】　4次元配列

下のデータを4次元配列に代入し，次に，年，地区，商品名を入力して，その年のその地区のその商品の売上数量，金額，利益および1人当たり売上を表示するプログラムをつくれ．

年別地区別商品別売上推移

年	地　区	商品名	数　量	金　額（万円）	利　益（万円）	1人当たり売　上
1987	東　京	パソコン	100	5000	1000	500
		ワープロ	70	1400	210	470
	大　阪	パソコン	70	3200	420	530
		ワープロ	60	1200	170	400
	名古屋	パソコン	60	2980	310	600
		ワープロ	30	600	90	600
1988	東　京	パソコン	150	7400	1420	740
		ワープロ	90	1700	250	540
	大　阪	パソコン	100	4800	550	800
		ワープロ	70	1300	160	650
	名古屋	パソコン	80	3560	410	710
		ワープロ	40	750	100	750

【解　説】　◎　4次元の配列を扱います.

① 表を配列に格納するためには，第1要素は1987年と1988年の2，第2要素は東京，大阪，名古屋の3，第3要素はパソコンとワープロの2，第4要素は数量，金額，利益，1人当たり売上の4です．したがって，次のように宣言します．ただし，

```
DIM h%(1 TO 2,1 TO 3,1 TO 2,1 TO 4)
```

これはOPTION BASE 1とすればDIM h%(2,3,2,4)と同じです.

② h%(2,3,1,2)は1988年の名古屋のパソコンの金額を示します.

③ データの読み込みは次のようにします.

```
FOR i=1 TO 2
      FOR j=1 TO 3
            FOR k=1 TO 2
                  FOR l=1 TO 4
                        READ h%(i,j,k,l)
NEXT l,k,j,i
```

【プログラム例】

```
DIM h%(1 TO 2, 1 TO 3, 1 TO 2, 1 TO 4)——4次元配列の宣言
FOR i = 1 TO 2 ————————————————————┐—データの読み込み
   FOR j = 1 TO 3                  │
      FOR k = 1 TO 2               │
         FOR l = 1 TO 4            │
            READ h%(i, j, k, l)    │
NEXT l, k, j, i ———————————————————┘
INPUT "ﾈﾝ "; y% ———————————————————┐—年，地区，商品名の入力
INPUT "ﾁｸ "; t$                    │
INPUT "ｼｮｳﾋﾝ "; s$ ————————————————┘
IF y% = 1987 THEN i = 1 ELSE i = 2———年が1987のときiを1，その他はiを2とする
IF t$ = "ﾄｳｷｮｳ" THEN———————————————┐—地区が東京のときjを1，大阪のときjを2,
   j = 1                           │  その他のときjを3とする
ELSEIF t$ = "ｵｵｻｶ" THEN            │
                        j = 2      │
                     ELSE          │
                        j = 3      │
END IF ————————————————————————————┘
IF s$ = "ﾊﾟｿｺﾝ" THEN k = 1 ELSE k = 2——商品名がパソコンのときkを1，その他のときkを2とする
FOR m = 1 TO 4 ————————————————————┐—i，j，kを各要素の値として数量，金額，利益，1人当たり
   PRINT h%(i, j, k, m);           │  売上を表示
NEXT m ————————————————————————————┘
END
DATA 100,5000,1000,500,70,1400,210,470
DATA 70,3200,420,530,60,1200,170,400
DATA 60,2980,310,600,30,600,90,600
DATA 150,7400,1420,740,90,1700,250,540
DATA 100,4800,550,800,70,1300,160,650
DATA 80,3560,410,710,40,750,100,750
```

【結　果】

```
ネン ? 1988                    1988，オオサカ，パソコンを入力の例
チク ? オオサカ
ショウヒン ? パソコン
 100   4800   550   800

ネン ? 1987                    1987，トウキョウ，ワープロを入力の例
チク ? トウキョウ
ショウヒン ? ワープロ
 70   1400   210   470
```

〔演習問題〕

(64)　配列 a (20) に下のデータを読み，2行に表示するプログラムをつくれ.

データ;3.5，2.5，1.5，3.3，4.2，5.2，2.6，3.5，4.2，
8.1，1.5，2.3，5.6，8.2，9.9，5.4，6.3，2.2，
1.8，4.4

(65)　x を-10 から 10 まで $y=3x^3-2x^2+5x$ の値を求めて配列 y%(-10)〜y%(10)に代入してから，各値を表示するプログラムをつくれ.

(66)　下のデータを配列に読み a%(5)と a%(7)を表示するプログラムをつくれ．ただし，配列の添字は1からとるものとする.

データ;7，5，2，9，7，3，8，4，1，0

(67)　オプションベースを1として，配列 x!(10,20)を宣言し，縦方向 10，横方向 20 の表と考え乱数をつくって代入し，x!(5,12)の位置の値を表示するプログラムをつくれ.

(68)　下の表を2次元配列に読み表示するプログラムをつくれ.

5	3	4
2	3	8
7	4	5
6	1	8

(69)　下の表の相対する項目の積の値を配列に代入して，表示するプログラムをつくれ.

8	7	6
2	9	3
4	8	2
7	1	0

7	4	9
5	3	1
6	8	2
3	2	7

(70)　【例題 61】のデータを使い，配列に値を読み込んでから，5品目の金額の合計を求めるプログラムをつくれ.

(71)　乱数で 100 個の 10〜99 の値をつくり，10〜90 台が各々が何回おきたかを求めるプログラムをつくれ.

(72)　下の文字データを配列に読み，後から表示するプログラムをつくれ.

データ;Komoro, Karuizawa, Nagano, Ueda, Ueno

(73)　(72) の文字列配列を使って，先頭1文字を入力して，その文字で始まる文字列を表示するプログラムをつくれ.

(74)　【例題 62】のデータを使い3次元配列に値を代入して，商品番号1または2を入力し，1のときは"パソコン"のデータ，2のときは"ワープロ"のデータを表示するプログラムをつくれ.

(75)　【例題 62】のデータを使い3次元配列に値を代入してから金額を入力し，その金額以上の利益の地区と品名のデータを表示するプログラムをつくれ.

〔解　答〕

(64)　**【プログラム例】**

```
DIM a(20) ──────────────── 配列 a (20) の宣言
FOR i = 0 TO 19 ─────────┐─ a(0)～a(19)にデータを読み込む
   READ a(i)             │
NEXT i ──────────────────┘
FOR i = 0 TO 1 ──────────┐─ a(0)～a(19)を2行にわけて表示
   FOR j = 0 TO 9        │
      PRINT a(10 * i + j);│  a(0)～a(9)が1行目, a(10)～a(19)が2行目
NEXT j                   │
PRINT                    │
NEXT i ──────────────────┘
END
DATA 3.5,2.5,1.5,3.3,4.2,5.2,2.6,3.5,4.2,8.1
DATA 1.5,2.3,5.6,8.2,9.9,5.4,6.3,2.2,1.8,4.4
```

【結　果】

```
3.5  2.5  1.5  3.3  4.2  5.2  2.6  3.5  4.2  8.1
1.5  2.3  5.6  8.2  9.9  5.4  6.3  2.2  1.8  4.4
```

(65)　**【プログラム例】**

```
DIM y%(-10 TO 10) ─────────────────── 配列 y%(-10)～y%(10)の宣言
FOR x = -10 TO 10 ──────────────────┐─ x を-10 から 10 まで代入して
   y%(x) = 3 * x * x * x - 2 * x * x + 5 * x │ y=3x²-2x²+5x を求めて配列へ代入
NEXT x ─────────────────────────────┘
FOR x = -10 TO 10 ──────────────────┐─ y%(-10)～y%(10)の表示
   PRINT y%(x)                      │
NEXT x ─────────────────────────────┘
END
```

【結　果】

```
-3250
-2394
-1704
-1162
-750
```

```
-450
-244
-114
-42
-10
 0
 6
 26
 78
 180
 350
 606
 966
 1448
 2070
 2850
```

(66) **【プログラム例】**

```
OPTION BASE 1 ──────────── 添字を1から始める
DIM a%(10) ─────────────── 配列 a%(1)～a%(10)の宣言
FOR i = 1 TO 10 ─────────┐ a%(1)～a%(10)にデータを読み込む
    READ a%(i)           │
NEXT i ──────────────────┘
PRINT a%(5); a%(7) ──────── a%(5) と a%(7)の表示
END
DATA 7,5,2,9,7,3,8,4,1,0
```

【結　果】

```
7  8
```

(67) **【プログラム例】**

```
OPTION BASE 1 ──────────── オプションベースを1とする
DIM x!(10, 20) ─────────── 配列 x!(10,20)の宣言
FOR i = 1 TO 10 ─────────┐ x!(1,1)から x!(10,20)に乱数を代入
    FOR j = 1 TO 20      │
        x!(i, j) = RND(1)┘
NEXT j, i
PRINT x!(5, 12) ────────── x!(5,12)の値を表示
END
```

【結　果】

```
.46298
```

(68) **【プログラム例】**

```
DIM x%(4, 3) ───────────── 2次元配列の宣言
FOR i = 0 TO 3 ──────────┐ 配列 x%にデータを読み込む
    FOR j = 0 TO 2       │
        READ x%(i, j)    │
NEXT j, i ───────────────┘
```

```
FOR i = 0 TO 3 ─────────────────┐表示
    FOR j = 0 TO 2              │
        PRINT x%(i, j);         │
    NEXT j                      │
PRINT                           │
NEXT i ─────────────────────────┘
END
DATA 5,3,4,2,3,8,7,4,5,6,1,8
```

【結　果】

```
5  3  4
2  3  8
7  4  5
6  1  8
```

(69) **【プログラム例】**

```
DIM s%(4, 3), t%(4, 3), u%(4, 3) ───────配列の宣言
FOR i = 0 TO 3 ─────────────────────────┐2つの表を配列s%, t%へ読み込み相対する値の
    FOR j = 0 TO 2                      │積を求めて配列u%の相対する項へ代入
        READ s%(i, j), t%(i, j)         │
        u%(i, j) = s%(i, j) * t%(i, j)  │
    NEXT j                              │
NEXT i ─────────────────────────────────┘
FOR i = 0 TO 3 ─────────────────────────┐結果の表示
    FOR j = 0 TO 2                      │
        PRINT USING "###    "; s%(i, j);│
    NEXT j                              │
    PRINT "    ";                       │
    FOR j = 0 TO 2                      │
        PRINT USING "###    "; t%(i, j);│
    NEXT j                              │
    PRINT "    ";                       │
    FOR j = 0 TO 2                      │
        PRINT USING "###    "; u%(i, j);│
    NEXT j                              │
PRINT                                   │
NEXT i ─────────────────────────────────┘
END
DATA 8,7,7,4,6,9,2,5,9,3,3,1,4,6,8,8,2,2,7,3,1,2,0,7
```

【結　果】

```
  8    7    6        7    4    9       56   28   54
  2    9    3        5    3    1       10   27    3
  4    8    2        6    8    2       24   64    4
  7    1    0        3    2    7       21    2    0
```

(70) **【プログラム例】**

```
DIM s$(5, 4) ───────────────────────────配列の宣言
FOR i = 0 TO 4 ─────────────────────────データの読み込みと3列目の計算を
    FOR j = 0 TO 2                       して代入および合計を求める
```

```
      READ s$(i, j)
    NEXT j
    s$(i, 3) = STR$(VAL(s$(i, 1)) * VAL(s$(i, 2)))
    t = t + VAL(s$(i, 3)) ──────────────┐金額の合計を求める
NEXT i ─────────────────────────────────┘
PRINT t ────────────────────────────────結果の表示
END
DATA ﾊﾟｿｺﾝ,25,458,ﾜｰﾌﾟﾛ,48,216,CD,21,142
DATA VTR,30,98,ﾋﾞﾃﾞｵｶﾒﾗ,13,126
```

【結　果】

```
29378
```

(71) 【プログラム例】

```
DIM s%(1 TO 9)──────────────────配列の宣言
FOR i = 1 TO 100────────────────┬乱数をつくり数える
    p = INT(RND(3) * 90) + 10───┼10～99の乱数
    pp = INT(p / 10)────────────┼乱数の10の値をとり出す
    s%(pp) = s%(pp) + 1─────────┼乱数の値によってs%(1)～s%(9)に代入
NEXT i ─────────────────────────┘
FOR i = 1 TO 9 ─────────────────┬結果の表示
PRINT i * 10; "ﾀﾞｲ="; s%(i)     │
NEXT i ─────────────────────────┘
END
```

【結　果】

```
10 ﾀﾞｲ= 14
20 ﾀﾞｲ= 4
30 ﾀﾞｲ= 16
40 ﾀﾞｲ= 13
50 ﾀﾞｲ= 9
60 ﾀﾞｲ= 12
70 ﾀﾞｲ= 10
80 ﾀﾞｲ= 9
90 ﾀﾞｲ= 13
```

(72) 【プログラム例】

```
DIM a$(1 TO 5) ─────────────────配列の宣言
FOR i = 1 TO 5──────────────────┬データの読み込み
    READ a$(i)                  │
NEXT i ─────────────────────────┘
FOR i = 5 TO 1 STEP -1──────────┬逆に表示
    PRINT a$(i); "  ";          │
NEXT i ─────────────────────────┘
PRINT
END
DATA Komoro,Karuizawa,Nagano,Ueda,Ueno
```

【結　果】

```
Ueno  Ueda  Nagano  Karuizawa  Komoro
```

(73) **【プログラム例】**

```
DIM a$(1 TO 5) ―――――――――――――――――――――― 配列の宣言
FOR i = 1 TO 5 ――――――――――――――――――――┐ データの読み込み
    READ a$(i)                              │
NEXT i ―――――――――――――――――――――――――――┘
INPUT x$ ――――――――――――――――――――――――――― 文字列の入力
FOR i = 1 TO 5 ――――――――――――――――――――┐ x$で始まる文字列の表示
    IF ASC(a$(i)) = ASC(x$) THEN PRINT a$(i); "  ";
NEXT i ―――――――――――――――――――――――――――┘
PRINT
END
DATA Komoro, Karuizawa, Nagano, Ueda, Ueno
```

【結　果】

```
? U
Ueda  Ueno
? K
Komoro  Karuizawa
```

(74) **【プログラム例】**

```
DIM r%(3, 2, 4) ――――――――――――――――――― 配列の宣言
FOR i = 0 TO 2 ――――――――――――――――――――┐ データの読み込み
    FOR j = 0 TO 1                          │
        FOR k = 0 TO 3                      │
            READ r%(i, j, k)                │
        NEXT k                              │
    NEXT j                                  │
NEXT i ―――――――――――――――――――――――――――┘
INPUT "ｼｮｳﾋﾝ NO (1 or 2) ", t% ―――――――― 商品番号の入力
IF t% = 1 THEN s$ = "ﾊﾟｿｺﾝ" ―――――――――― 商品番号によって s$を決める
IF t% = 2 THEN s$ = "ﾜｰﾌﾟﾛ"
c% = t% - 1 ――――――――――――――――――――――― 商品番号によって商品を示す c%を決める
t1$(0) = "ﾄｳｷｮｳ "――――――――――――――――┐ 地区名を配列へ代入
t1$(1) = "ｵｵｻｶ  "                      │
t1$(2) = "ﾅｺﾞﾔ  " ―――――――――――――――――┘
PRINT s$ ――――――――――――――――――――――――――┐ 結果の表示
PRINT "         ｽｳﾘｮｳ   ｷﾝｶﾞｸ   ﾘｴｷ   ｳﾘｱｹﾞ/ﾆﾝ"
FOR i = 0 TO 2
    PRINT t1$(i);
    PRINT r%(i, c%, 0); "  "; r%(i, c%, 1); "  "; r%(i, c%, 2); r%(i, c%, 3)
NEXT i ―――――――――――――――――――――――――――┘
END
DATA 350, 500, 170, 100, 650, 800, 120, 180
DATA 250, 300, 110, 120, 400, 500, 100, 110
DATA 210, 230, 104, 115, 350, 450, 105, 109
```

【結　果】

```
ｼｮｳﾋﾝ NO (1 or 2) 1                        1を入力の例
ﾊﾟｿｺﾝ
         ｽｳﾘｮｳ   ｷﾝｶﾞｸ   ﾘｴｷ   ｳﾘｱｹﾞ/ﾆﾝ
ﾄｳｷｮｳ    350     500     170    100
```

```
オオサカ　　250　　300　　110　120
ナゴヤ　　　210　　230　　104　115
ショウヒン NO（1 or 2）2                 2を入力の例
ワープロ
　　　　　スウリョウ　キンガク　リエキ　ウリアゲ/ニン
トウキョウ　650　　800　　120　180
オオサカ　　400　　500　　100　110
ナゴヤ　　　350　　450　　105　109
```

(75) **【プログラム例】**

```
DIM r%(3, 2, 4) ──────────────── 配列の宣言
FOR i = 0 TO 2 ──────────────┐── データの読み込み
    FOR j = 0 TO 1
        FOR k = 0 TO 3
            READ r%(i, j, k)
        NEXT k
    NEXT j
NEXT i ──────────────────────┘
INPUT "キンガク ", k% ──────────── 金額の入力
t1$(0) = "トウキョウ "──────────┐── 地区名を配列へ代入
t1$(1) = "オオサカ  "
t1$(2) = "ナゴヤ   " ──────────┘
h$(0) = "パソコン" ─────────────┐── 商品名を配列へ代入
h$(1) = "ワープロ" ─────────────┘
PRINT k%; "イジョウ"──────────────────────────┐── 結果の表示
PRINT "チク     ヒンメイ  リエキ"
FOR i = 0 TO 2
    IF r%(i, 0, 2) >= k% THEN PRINT t1$(i); h$(0); r%(i, 0, 2)
    IF r%(i, 1, 2) >= k% THEN PRINT t1$(i); h$(1); r%(i, 1, 2)
NEXT i ──────────────────────────────────────┘
END
DATA 350,500,170,100,650,800,120,180
DATA 250,300,110,120,400,500,100,110
DATA 210,230,104,115,350,450,105,109
```

【結　果】

```
キンガク 110                  110を入力の例
 110 イジョウ
チク　　　ヒンメイ　リエキ
トウキョウ パソコン 170
トウキョウ ワープロ 120
オオサカ　 パソコン 110
キンガク 105                  105を入力の例
 105 イジョウ
チク　　　ヒンメイ　リエキ
トウキョウ パソコン 170
トウキョウ ワープロ 120
オオサカ　 パソコン 110
ナゴヤ　　 ワープロ 105
キンガク 130                  130を入力の例
 130 イジョウ
チク　　　ヒンメイ　リエキ
トウキョウ パソコン 170
```

9章　タイプ型

TYPE, END TYPE

【例題 64】　ユーザー定義型

下のデータをユーザー定義型（タグを KOZIN とし，名前を 10 文字，年齢を整数型，年収を単精度実数型，出身地を 7 文字とするメンバー）で定義し，各項名を表示するプログラムをつくれ.

データ;Onoda, 50, 1350.6, トウキョウト

（名前，年齢，年収（万円），出身地の順とする）

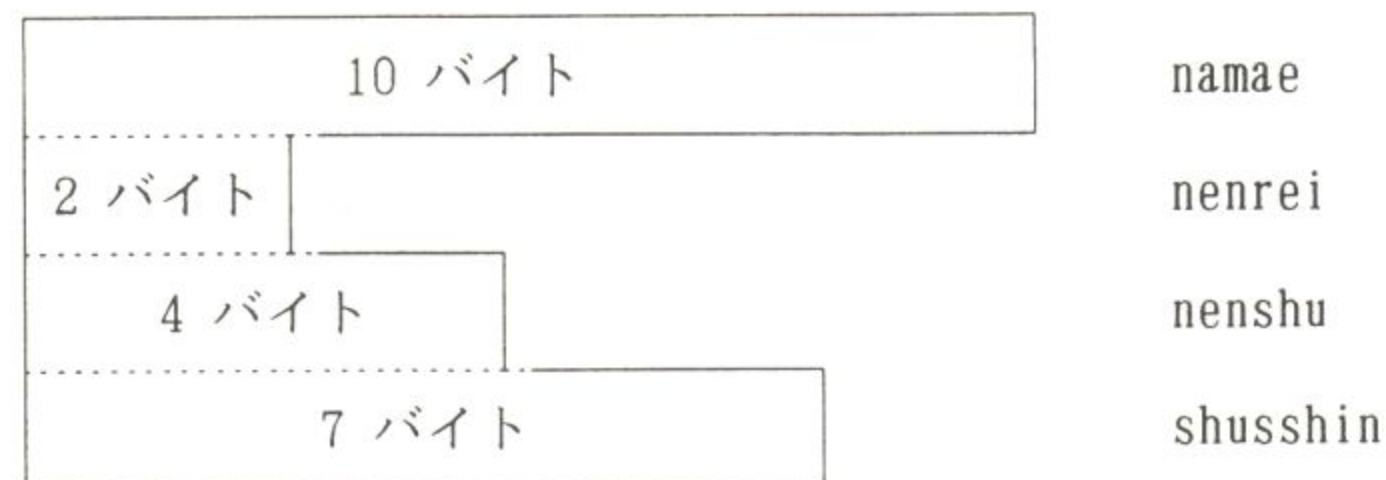
【解　説】　◎　ユーザー定義型（データ型）を使います.

① ユーザー定義型は C 言語の構造型に相当するもので，従来の BASIC にはない新しいデータ型です. 本例では名前，年齢，年収，出身地の組合わせのデータを 1 つのかたまりとして扱おうとするものです.

② ユーザー定義型は次のように定義します.

```
TYPE kozin                      ……全体を kozin という名前とします. タグと呼びます.
   namae AS STRING*10           ……namae をメンバー名とし，固定長 10 文字とします      ┐
   nenrei AS INTEGER            ……nenrei をメンバー名とし，整数型とします.          │
   nenshu AS SINGLE             ……nenshu をメンバー名とし，単精度実数とします       │ メンバー
   shusshin AS STRING*7         ……shusshin をメンバー名とし，固定長 7 文字とします  ┘
END TYPE                        ……ユーザー定義型の定義の終了
```

この結果次のような 1 かたまりの領域が確保されます. 文字型，整数型，実数型，文字型と型の違うものを 1 かたまりとして扱います.

10 バイト	namae
2 バイト	nenrei
4 バイト	nenshu
7 バイト	shusshin

③　このタグ kozin の変数は次のように宣言します.

```
  ┌── レコード型変数名
DIM a AS kozin
```

④　各メンバーを参照（各メンバーに値を代入したり，読み出したりすること）するには次のように行います.

```
a.namae="Onoda"
a.nenrei=50
a.nenshu=1350.6
a.shusshin="トウキョウ"
```

⑤　表示は PRINT a.namae で"Onoda"を表示します.

⑥　メンバーは配列と可変型の文字列（たとえば，x$という型．これは x$の内容が"ABC"と 3 文字であったり"ABCDE"と 5 文字であったりします）にはなれません.

【プログラム例】

```
TYPE kozin ──────────────── ユーザー定義型の宣言
    namae AS STRING * 10 ──── メンバーnamae は 10 文字の文字列
    nenrei AS INTEGER ─────── メンバーnenri は整数
    nenshu AS SINGLE ──────── メンバーnenshu は実数
    shusshin AS STRING * 7 ── メンバーshusshin は 7 文字の文字列
END TYPE
DIM a AS kozin ───────────── 変数名 a の宣言
a.namae = "Onoda" ────────── 変数にデータを入力
a.nenrei = 50
a.nenshu = 1350.6
a.shusshin = "ﾄｳｷｮｳﾄ"
PRINT a.namae ────────────── 各変数の表示
PRINT a.nenrei
PRINT a.nenshu
PRINT a.shusshin
END
```

【結　果】

```
Onoda
 50
 1350.6
ﾄｳｷｮｳﾄ
```

【例題 65】　ユーザー定義型での配列

下表のデータをタグを shain としてユーザー定義型で定義して，配列にデータを読み，全データを表示するプログラムをつくれ.

名　前	年　収	家族数
Yamada	1500	6
Tanaka	1800	4
Onoda	850	3
Kawada	1200	4
Noda	1000	5
Hoshino	750	3

【解　説】　◎　変数に配列を使います.

① データ組が複数の場合は変数に配列を使います.

② 【例題 64】と同様にタグを shain としてメンバーを namae，nenshu，kazokusu として定義します.

③ 変数の定義を次のようにします. この結果，0～5 の 6 組のデータを扱えます.

DIM a(6)　AS shain

④ 参照は a(1).namae，a(3).nenshu，a(4).kazokusu のように配列名.メンバー名で行います.

【プログラム例】

```
TYPE shain ──────────────────┐ユーザー定義型の定義
    namae AS STRING * 10
    nenshu AS INTEGER
    kazokusu AS INTEGER
END TYPE ────────────────────┘
DIM a(6) AS shain ─────────── 変数の宣言
FOR i = 0 TO 5 ──────────────┐データの読み込み
    READ a(i).namae
    READ a(i).nenshu
    READ a(i).kazokusu
NEXT i ──────────────────────┘
FOR i = 0 TO 5 ──────────────────────────────┐表示
    PRINT USING "&          &"; a(i).namae;
    PRINT USING "#####  ###"; a(i).nenshu; a(i).kazokusu
NEXT i ──────────────────────────────────────┘
END
DATA Yamada,1500,6,Tanaka,1800,4,Onoda,850,3
DATA Kawada,1200,4,Noda,1000,5,Hoshino,750,3
```

【結　果】

```
Yamada      1500    6
Tanaka      1800    4
Onoda        850    3
Kawada      1200    4
Noda        1000    5
Hoshino      750    3
```

【例題 66】　タイプ型を使って検索

次のデータをタイプ型で定義して代入し，次に，身長を入力してその身長以上の人のリストをつくるプログラムをつくれ.

名　前	身長（cm）	体重（kg）	年　齢
神田国男	183.7	65.8	41
大友良雄	160.6	54.2	28
中根富夫	158.9	50.6	36
井上太郎	190.3	72.8	29
吉田　薫	181.3	83.9	33
渡辺兼介	175.2	59.2	42
新井久男	170.6	70.3	38

【解　説】　◎　タイプ型変数を使って検索をします.

① 問題のデータを次の名前でタイプ型変数に代入します.

```
TYPE people
      namae AS STRING*20          名前20バイト
      shinchou AS SINGLE          身長　単精度実数
      taiju AS SINGLE             体重　単精度実数
      nenrei AS INTEGER           年齢　整数
END TYPE
```

② データを配列 a(i) に代入します.

③ 身長を shi に代入して，変数名 a(i).shinchou と比較して，shi 以上の人のリストを表示します.

【プログラム例】

```
TYPE people ─────────────┐─タイプ型変数の宣言
    namae AS STRING * 20       │ 名前，身長，体重，年齢を代入する準備
    shinchou AS SINGLE         │
    taiju AS SINGLE            │
    nenrei AS INTEGER          │
END TYPE ────────────────┘
DIM a(7) AS people ──────────── 配列変数の宣言
```

```
FOR i = 0 TO 6 ─────────┐配列へデータ読み込み
    READ a(i).namae      │
    READ a(i).shinchou   │
    READ a(i).taiju      │
    READ a(i).nenrei     │
NEXT i ─────────────────┘
INPUT "ｼﾝﾁｮｳ ", shi ──────────────────身長入力
FOR i = 0 TO 6 ──────────────────────┐shi 以上の人の名前を表示
    IF a(i).shinchou >= shi THEN PRINT a(i).namae │180.6 を入力の例
NEXT i ──────────────────────────────┘
END
DATA ｶﾝﾀﾞ ｸﾆｵ,183.7,65.8,41
DATA ｵｵﾄﾓ ﾖｼｵ,160.6,54.2,28
DATA ﾅｶﾈ ﾄﾐｵ,158.9,50.6,36
DATA ｲﾉｳｴ ﾀﾛｳ,190.3,72.8,29
DATA ﾖｼﾀﾞ ｶｵﾙ,181.3,83.9,33
DATA ﾜﾀﾅﾍﾞ ｹﾝｽｹ,175.2,59.2,42
DATA ｱﾗｲ ﾋｻｵ,170.6,70.3,38
```

【結　果】

```
ｼﾝﾁｮｳ 180.6
ｶﾝﾀﾞ ｸﾆｵ
ｲﾉｳｴ ﾀﾛｳ
ﾖｼﾀﾞ ｶｵﾙ
```

【例題 67】　全体のコピー

次のデータ組を，タグを shain とするユーザー定義型で定義し，変数 a と b を宣言し，a.namae，a.nenshu，a.kazokusu に代入した後，b へコピーして，b の各メンバーの値を表示するプログラムをつくれ.

データ：

yamada	1500	6

名前，年収，家族数とする

【解　説】　◎　ユーザー定義型の全データを変数間でコピーします.

① タグを shain とし，メンバーを namae，nenshu，kazokusu として定義します.

② 次に変数 a と b を宣言します.

③ 各変数にデータを読み込みます.

④ b=a によって a のすべてのメンバーを b に代入します．ユーザー定義型のすべてのデータを変数間でコピーできる便利さがあります.

【プログラム例】

```
TYPE shain ─────────────────┐ユーザー定義型の宣言
    namae AS STRING * 10     │
    nenshu AS INTEGER        │
    kazokusu AS INTEGER      │
END TYPE ───────────────────┘
DIM a AS shain ─────────────┐変数 a，b の宣言
DIM b AS shain ─────────────┘
```

```
READ a.namae ─────────────────── 変数aへデータを読み込む
READ a.nenshu
READ a.kazokusu
b = a ──────────────────────────── aをbへコピー
PRINT USING "&          &"; b.namae; ───── bの表示
PRINT USING "#####  ###"; b.nenshu; b.kazokusu
END
DATA Yamada,1500,6
```

【結　果】

```
Yamada        1500      6
```

〔演習問題〕

(76)　タグをkaishaとするユーザー定義型でshamei，uriage，jugyoinをメンバーとして定義して，次に変数名をbとして，下のデータを代入してから表示するプログラムをつくれ．

データ;Nihon Seisakusho，1536.2，1200（社名，売上（億円），従業員（人））の順とする．

(77)　下のデータ組を，ユーザー定義型で定義し，変数aとbを宣言し，aにデータ組を代入し，次に全体をbへコピーしてからbを表示するプログラムをつくれ．

データ：

大山	東京都	3

名前，住所，家族数とする

(78)　次のデータを，ユーザー定義型で定義し，a(4)を宣言して代入した後，全体を表示するプログラムをつくれ．

大山	東京都	3
上田	埼　玉	4
飯田	千　葉	5
豊島	神奈川	4

(79)　下のデータをユーザー定義型で定義して，次に名前を入力して，その名前の人のデータを表示するプログラムをつくれ．

名　前	年齢	所　属	特　技	勤続年数	身長（cm）	体重（kg）
大山和男	31	営業1課	英会話	7	163.8	60.3
木下三郎	28	総務課	パソコン	3	156.9	51.4
上田次郎	19	営業3課	音　楽	1	172.8	62.5
玉井久雄	24	経理課	英会話	2	178.3	70.6
川上太郎	33	営業2課	パソコン	5	181.5	75.3
大下一夫	25	企画課	書　道	4	170.2	63.6

(80)　(79) のデータを使って勤続年数を入力し，その年数以上の人のリストを表示するプログラムをつくれ.

〔解　答〕

(76) **【プログラム例】**

```
TYPE kaisha ──────────────────── ユーザー定義型の宣言
    shamei AS STRING * 20
    uriage AS SINGLE
    jugyoin AS INTEGER
END TYPE
DIM b AS kaisha ──────────────── 変数名を b とする
b.shamei = "Nihon Seisakusho" ─── 代入
b.uriage = 1536.2
b.jugyoin = 1200
PRINT b.shamei ───────────────── 表示
PRINT b.uriage; "ｵｸｴﾝ"
PRINT b.jugyoin; "ﾆﾝ"
END
```

【結　果】

```
Nihon Seisakusho
 1536.2 ｵｸｴﾝ
 1200 ﾆﾝ
```

(77) **【プログラム例】**

```
TYPE meibo ───────────────────── ユーザー定義型の宣言
    namae AS STRING * 10
    jusho AS STRING * 8
    kazokusu AS INTEGER
END TYPE
DIM a AS meibo ───────────────── 変数 a，b の宣言
DIM b AS meibo
READ a.namae ─────────────────── 変数に値を読み込む
READ a.jusho
READ a.kazokusu
b = a ────────────────────────── b に a をコピー
    PRINT b.namae; b.jusho; b.kazokusu ── 表示
END
DATA ｵｵﾔﾏ,ﾄｳｷｮｳﾄ,3
```

【結　果】

```
ｵｵﾔﾏ          ﾄｳｷｮｳﾄ     3
```

(78) 【プログラム例】

```
TYPE meibo ──────────────────── ユーザー定義型の宣言
    namae AS STRING * 10
    jusho AS STRING * 8
    kazokusu AS INTEGER
END TYPE
DIM a(4) AS meibo ───────────── 変数名をa(4)と配列とする
FOR i = 0 TO 3 ──────────────── 変数に値を読み込む
    READ a(i).namae
    READ a(i).jusho
    READ a(i).kazokusu
NEXT i
FOR i = 0 TO 3 ──────────────── 表示
    PRINT a(i).namae; a(i).jusho; a(i).kazokusu
NEXT i
END
DATA ｵｵﾔﾏ,ﾄｳｷｮｳﾄ,3,ｳｴﾀﾞ,ｻｲﾀﾏ,4
DATA ｲｲﾀﾞ,ﾁﾊﾞ,5,ﾄｼﾏ,ｶﾅｶﾞﾜ,4
```

【結　果】

```
ｵｵﾔﾏ        ﾄｳｷｮｳﾄ    3
ｳｴﾀﾞ        ｻｲﾀﾏ      4
ｲｲﾀﾞ        ﾁﾊﾞ       5
ﾄｼﾏ         ｶﾅｶﾞﾜ     4
```

(79) 【プログラム例】

```
TYPE shain ──────────────── タイプ型変数宣言
                            名前，年齢，所属，特技，勤続，身長，体重
    namae AS STRING * 8
    nenrei AS INTEGER
    shozoku AS STRING * 10
    tokugi AS STRING * 10
    kinzoku AS INTEGER
    shinchou AS SINGLE
    taiju AS SINGLE
END TYPE
DIM a(6) AS shain ───────── 配列の宣言
FOR i = 0 TO 5 ──────────── データを配列へ読み込む
    READ a(i).namae
    READ a(i).nenrei
    READ a(i).shozoku
    READ a(i).tokugi
    READ a(i).kinzoku
    READ a(i).shinchou
    READ a(i).taiju
NEXT i
INPUT "名前  ", na$ ─────── 名前の入力
FOR i = 0 TO 5 ──────────────────────────── na$と同じ名前の
    IF a(i).namae = na$ THEN                人のデータの表示
        PRINT a(i).namae; a(i).nenrei; a(i).shozoku; a(i).tokugi;
        PRINT a(i).kinzoku; a(i).shinchou; a(i).taiju
    END IF
```

```
NEXT i
END
DATA 大山和男,31,営業一課,英会話,7,163.8,60.3
DATA 木下三郎,28,総務課,パソコン,3,156.9,51.4
DATA 上田次郎,19,営業三課,音楽,1,172.8,62.5
DATA 玉井久雄,24,経理課,英会話,2,178.3,70.6
DATA 川上太郎,33,営業二課,パソコン,5,181.5,75.3
DATA 木下一夫,25,企画課,書道,4,170.2,63.6
```

【結　果】

```
名前　木下三郎
木下三郎 28 総務課      パソコン    3  156.9  51.4      木下三郎を入力の例
```

(80) **【プログラム例】**

```
TYPE shain ──────────────────────┐タイプ型
    namae AS STRING * 8           │名前，年齢，所属，特技，勤続，身長，体重
    nenrei AS INTEGER             │
    shozoku AS STRING * 10        │
    tokugi AS STRING * 10         │
    kinzoku AS INTEGER            │
    shinchou AS SINGLE            │
    taiju AS SINGLE               │
END TYPE ────────────────────────┘
DIM a(6) AS shain ───────────────── 配列宣言
FOR i = 0 TO 5 ──────────────────┐データの読み込み
    READ a(i).namae               │
    READ a(i).nenrei              │
    READ a(i).shozoku             │
    READ a(i).tokugi              │
    READ a(i).kinzoku             │
    READ a(i).shinchou            │
    READ a(i).taiju               │
NEXT i ──────────────────────────┘
INPUT "勤続年数  ", ne ─────────── 勤続年数の入力
FOR i = 0 TO 5 ──────────────────────────┐勤続年数が ne 以上の人をリスト
    IF a(i).kinzoku >= ne THEN PRINT a(i).namae │
NEXT i ──────────────────────────────────┘
END
DATA 大山和男,31,営業一課,英会話,7,163.8,60.3
DATA 木下三郎,28,総務課,パソコン,3,156.9,51.4
DATA 上田次郎,19,営業三課,音楽,1,172.8,62.5
DATA 玉井久雄,24,経理課,英会話,2,178.3,70.6
DATA 川上太郎,33,営業二課,パソコン,5,181.5,75.3
DATA 木下一夫,25,企画課,書道,4,170.2,63.6
```

【結　果】

```
勤続年数  4              4を入力の例
大山和男
川上太郎
木下一夫
```

10章　式の定義

DEF FN

【例題 68】　DEF

円の面積を求める式 3.14159*r*r（ただし r は半径）を定義して，次に半径 r を入力して，半径 r の円の面積を求めるプログラムをつくれ．

【解　説】　◎　DEF により式を定義します．

① DEF fna(r)=3.14159*r*r によって 3.14159*r*r を fna(r) という名前で定義します．fn の次の a が変数名で最大 40 文字まで使えます．

② ( )内の r がパラメータで r の値が与えられると 3.14159*r*r の値を fna(r) が持ちます．たとえば r を 5 とすると fna(5) は 3.14159×5×5 となります．

③ DEF FN 関数は定義されたモジュールの中でのみ使えます．他のモジュールからの参照はできません．

④ DEF fna(r) の型は単精度実数型です．DEF fna!(r) と明示することができます．

【プログラム例】

```
DEF fna (r) = 3.14159 * r * r────────式の定義
INPUT "r "; r ─────────────────────半径 r の入力
s = fna(r) ────────────────────────式の値を求め s に代入
PRINT "ﾊﾝｹｲ"; r; " ﾉ ﾒﾝｾｷ="; s ──────結果の表示
END
```

【結　果】

```
r ? 123                          ─────── 123 を入力の例
ﾊﾝｹｲ 123   ﾉ ﾒﾝｾｷ= 47529.12
r ? 5.8                          ─────── 5.8 を入力の例
ﾊﾝｹｲ 5.8   ﾉ ﾒﾝｾｷ= 105.6831
```

【例題 69】　複数のパラメータ

ax^2+bx+c を関数で定義し，次に a，b，c，x の値を代入して，ax^2+bx+c の値を求めるプログラムをつくれ．

【解　説】　◎　複数のパラメータを使います．

① DEF fna(a,b,c,x)=a*x*x+b*x+c では ax^2+bx+c を定義します．このときのパラメータは a，b，c，x の 4 つです．

【プログラム例】

```
DEF fna (a, b, c, x) = a * x * x + b * x + c ──────a²+bx+c の定義
INPUT "a,b,c,x "; a, b, c, x ───────────────a, b, c, x の入力
y = fna(a, b, c, x) ──────────────────────式の値を求め y に代入
IF b >= 0 THEN z1$ = " +" ELSE z1$ = " " ────┬─+, −のときの記号の表示のための処理
IF c >= 0 THEN z2$ = "+" ELSE z2$ = " " ─────┘
PRINT a; "*"; x; "^2"; z1$; b; "*"; x; z2$; c; "="; y──結果の表示
END
```

【結　果】

```
a,b,c,x ? 3,-5,2,3─────────── 3, −5, 2, 3, の入力の例
 3 * 3 ^2 -5 * 3 + 2 = 14
a,b,c,x ? 2,5,4,2──────────── 2, 5, 4, 2 を入力の例
 2 * 2 ^2 + 5 * 2 + 4 = 22
a,b,c,x ? -3,7,-6,2─────────── −3, 7, −6, 2 を入力の例
-3 * 2 ^2 + 7 * 2  -6 =-4
a,b,c,x ? 2,5,3,-4─────────── 2, 5, 3, −4 を入力の例
 2 *-4 ^2 + 5 *-4 + 3 = 15
```

【例題 70】　文字列を値とする関数式

"Microsoft□"+a$を文字列とする式を定義し，次に文字列を入力して，Microsoft とつないで表示するプログラムをつくれ．

【解　説】　◎　文字列を値とする関数式を定義します．

① DEF fnx$(a$)="Microsoft□"+a$の定義によって"Microsoft□"と文字列 a$をつないだ文字列を式 fnx$(a$) の値とします．

【プログラム例】

```
DEF fnx$ (a$) = "Microsoft " + a$ ────式の定義
INPUT x$ ─────────────────────x$の入力
y$ = fnx$(x$)──────────────────式の値を y$に代入
PRINT y$───────────────────────結果の表示
END
```

【結　果】

```
? Quick C────────────── Quick C を入力の例
Microsoft Quick C
? Quick Basic─────────── Quick Basic を入力の例
Microsoft Quick Basic
```

【例題 71】　関数式の演算

a^3 と b^2 を求める式を各々fnx(a)とfnb(b)で定義し，次に，x と y を入力し fnx(x)/fnb(y)の値を求めるプログラムをつくれ．

【解　説】　◎　関数式の演算を行います．

① 下の定義で関数 fnx と fnb を定義します．

```
DEF fnx(a)=a*a*a
DEF fnb(b)=b*b
```

② x と y を入力し，z=fnx(x)/fnb(y)を行うと x^3/y^2 を求めて z に代入します.

③ もちろん DEF fnx(a,b)=a*a*a/(b*b)によって一度に式を定義できますが a^3 と b^2 を別のところで使いたい場合は別々に定義する必要があります.

【プログラム例】

```
DEF fnx (a) = a * a * a ――――― a³ の式の定義
DEF fnb (b) = b * b ――――――― b² の式の定義
INPUT x, y ―――――――――――― x，y の入力
z = fnx(x) / fnb(y)――――――― x³/y² を求めて z に代入
PRINT x; "^3 /"; y; "^2="; z ―― 結果の表示
END
```

【結　果】

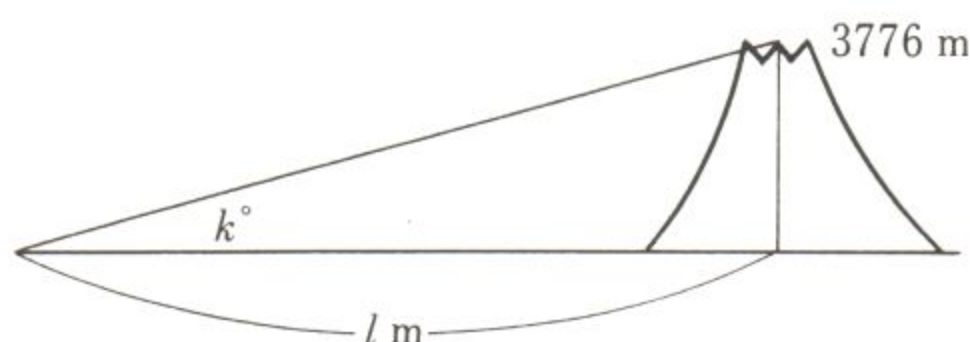
```
? 6.3,2.4 ――――――――――――― 6.3，2.4 を入力の例
 6.3 ^3 / 2.4 ^2= 43.41094
? 5,3 ――――――――――――――― 5，3 を入力の例
 5 ^3 / 3 ^2= 13.88889
? 4,2 ――――――――――――――― 4，2 を入力の例
 4 ^3 / 2 ^2= 16
```

【例題 72】　算術関数を使った関数式

下図の角度 k を入力して，距離 l を求めるプログラムをつくれ.

3776 m

k°

l m

【解　説】 ◎ 関数定義の中で算術関数を使います.

① 関数定義の中で，Quick BASIC の持つ関数を使うことができます．次の定義は l を求める式です．算術関数 TAN（正接）を使っています.

```
DEF fna(k)=3776/TAN(k/180*3.14159)
    ↑              ↑      ↑
```

- DEF fna(k)：k 度のときの l を求める式の定義
- TAN：正接
- k/180*3.14159：k 度をラジアンに変換

【プログラム例】

```
DEF fna (k) = 3776 / TAN(k / 180 * 3.14159)――― 式の定義
INPUT "k "; k ――――――――――――――――――――― k 度の入力
l = fna(k) ―――――――――――――――――――――― l を求める
PRINT "ｶｸ"; k; " ﾄﾞ ﾉ ﾄｷ ﾉ ｷｮﾘ="; l; "m"――――― 結果の表示
END
```

【結　果】

```
k ? 10――――――――――――――――――――――――――――10 を入力の例
ｶｸ 10  ﾄﾞ ﾉ ﾄｷ ﾉ ｷｮﾘ= 21414.78 m
k ? 30――――――――――――――――――――――――――――30 を入力の例
ｶｸ 30  ﾄﾞ ﾉ ﾄｷ ﾉ ｷｮﾘ= 6540.23 m
k ? 70――――――――――――――――――――――――――――70 を入力の例
ｶｸ 70  ﾄﾞ ﾉ ﾄｷ ﾉ ｷｮﾘ= 1374.356 m
```

〔演習問題〕

(81)　【例題 72】の図の距離 l を入力して，角 k 度を求めるプログラムをつくれ.

(82)　a，b を入力して$\sqrt{a^2+b^2}$を求めるプログラムをつくれ.

(83)　文字列を" で囲む式を定義し，文字列を入力して，" "で囲んで表示するプログラムをつくれ.

(84)　文字列から右端 2 文字をとり出した文字列をつくる式を定義し，文字列を入力して，右端 2 文字を表示するプログラムをつくれ.

(85)　文字列から左端 2 文字と右端 2 文字をとり出してつなげる式を定義し，次に文字列を入力して前 2 文字と後 2 文字をつないで表示するプログラムをつくれ.

(86)　"Microsoft"の a 番目から b 個の文字をとり出す式を定義し，次に数 x，y を入力して x 番目から y 個の文字をとり出して表示するプログラムをつくれ.

〔解　答〕

(81)　【プログラム例】

```
DEF fna (l) = ATN(3776 / l) / 3.14159 * 180―――k 度を求める式の定義
INPUT "l "; l ―――――――――――――――――――――――――l の入力
k = fna(l) ――――――――――――――――――――――――――k 度を求める
PRINT "ｷｮﾘ"; l; "m ﾉ ﾄｷ ﾉ ｶｸﾄﾞ="; k; "ﾄﾞ"――――結果の表示
END
```

【結　果】

```
l ? 21414 ―――――――――――――――――――――――――21414 を入力の例
ｷｮﾘ 21414 m ﾉ ﾄｷ ﾉ ｶｸﾄﾞ= 10.00036 ﾄﾞ
l ? 100000 ――――――――――――――――――――――――100000 を入力の例
ｷｮﾘ 100000 m ﾉ ﾄｷ ﾉ ｶｸﾄﾞ= 2.162463 ﾄﾞ
```

(82)　【プログラム例】

```
DEF fnx (a, b) = SQR(a * a + b * b)――― √(a²+b²)を求める式の定義
INPUT "a,b "; a, b ―――――――――――――――― a，b の入力
y = fnx(a, b) ――――――――――――――――――――― √(a²+b²)を求める
PRINT ; y ―――――――――――――――――――――――結果の表示
END
```

【結　果】

```
a,b ? 3,4 ――――― 3, 4 を入力の例
 5
a,b ? 25,19 ――――― 25, 19 を入力の例
 31.40064
```

(83)　【プログラム例】

```
DEF fnx$ (a$) = CHR$(34) + a$ + CHR$(34) ――― "文字列"の式の定義
INPUT x$ ――――――――――――――――――――――― 文字列の入力
y$ = fnx$(x$) ―――――――――――――――――――― 文字列を""で囲む
PRINT y$ ――――――――――――――――――――――― 結果の表示
END
```

【結　果】

```
? Quick Basic ――――――― Quick Basic を入力の例
"Quick Basic"
? ﾀﾉｼｲ ﾌﾟﾛｸﾞﾗﾑ ――――――― タノシイ　プログラムを入力の例
"ﾀﾉｼｲ ﾌﾟﾛｸﾞﾗﾑ"
```

(84)　【プログラム例】

```
DEF fnx$ (a$) = RIGHT$(a$, 2) ――― 右端 2 文字をとり出す式の定義
INPUT x$ ―――――――――――――――――― 文字列の入力
y$ = fnx$(x$) ――――――――――――――― 文字列から 2 文字をとり出す
PRINT y$ ―――――――――――――――――― 結果の表示
END
```

【結　果】

```
? picture ――――――― picture を入力の例
re
? computer ――――――― computer を入力の例
er
```

(85)　【プログラム例】

```
DEF fnx$ (a$) = LEFT$(a$, 2) + RIGHT$(a$, 2) ―― 左 2 文字と右 2 文字をとり出す式の定義
INPUT x$ ――――――――――――――――――――――――――― 文字列の入力
y$ = fnx$(x$) ―――――――――――――――――――――――― 前 2 文字と後 2 文字をとり出してつなぐ
PRINT y$ ――――――――――――――――――――――――――― 結果の表示
END
```

【結　果】

```
? giant ――――――― giant を入力の例
gint
? teacher ――――――― teacher を入力の例
teer
```

（86）【プログラム例】

```
DEF fnx$ (a, b) = MID$("Microsoft", a, b)——"Microsoft"の前からa番目からb個とり出す式の定義
INPUT x, y ————————————————————————x, yの入力
PRINT fnx$(x, y) ——————————————————"Microsoft"の前からx番目からy個とり出して表示
END
```

【結　果】

```
? 2,3————————2, 3を入力の例
icr
? 3,2————————3, 2を入力の例
cr
? 4,5————————4, 5を入力の例
rosof
```

11章　プロシジュア・サブルーチン

DECLARE SUB, SUB, END SUB, CALL

> **【例題 73】　プロシジュア・サブルーチン**
>
> メインプログラムで"Quick"と表示し，サブルーチン sa で"□BASIC"と続けて表示するプログラムをつくれ．

【解　説】　◎　プロシジュアのうちサブルーチンを使います．

① サブルーチンはメインプログラムとは別に作成して，メインプログラムから呼び出して何回でも使える便利な手法です．次の例のように，処理１を何回も使うとき，毎回その処理を記入せずに，サブルーチンとして別の場所にしまっておいて，呼び出して使えば，処理１は１回書けばよいことになります．

処理１を必要のつど記入したプログラム	処理１をサブルーチン化したプログラム
≀	≀
処理１	サブルーチンの呼び出し
≀	≀
処理１	サブルーチンの呼び出し
≀	≀
処理１	サブルーチンの呼び出し
≀	≀
(処理１の内容が 100 行になるような場合を考えるとこのプログラムの大変さがよくわかります)	サブルーチン 処理１

② 最初は簡単な例です．まずサブプログラムに名前をつけて宣言します．

```
DECLARE SUB sa()
```

DECLARE SUB サブプログラム名となります．サブプログラム名の後は( )をつけます．この意味は後で示しますが，サブプログラムへ引数を渡すことがあるからです．今回は引数はありません．

③ サブプログラムの呼び出しは CALL サブプログラム名またはサブプログラム名のみです．ただし，サブプログラム名のみでサブプログラムを呼び出すときは必ず DECLARE SUB を使ってサブプログラムの宣言をしておかねばなりません．CALL を使う場合は DECLARE SUB がなくても使えます．

④ サブプログラムは先頭に SUB サブプログラム名をつけてつくります．SUB プログラム名を入力す

ると（今回は SUB sa ⏎）次のように表示されますので，サブプログラムの内容をかきます．

```
SUB sa
END SUB
```

⑤ 本例ではメインプログラムで"Quick"を表示した後，サブプログラム sa を呼んで"□BASIC"を表示します．

```
DECLARE SUB sa( )        サブプログラム sa の宣言
  ≀
sa                       サブプログラム sa
                         を呼び出して実行
  ≀
END

SUB sa                   終了すると sa の次へもどる
  ≀
END SUB
```

【プログラム例】

```
DECLARE SUB sa () ――――― サブルーチン sa の宣言
PRINT "Quick"; ――――――― "Quick"の表示
sa
END ――――――――――――――― サブルーチン sa の呼び出し

SUB sa ―――――――――――――― サブルーチン sa
    PRINT " BASIC" ―――― "□BASIC"の表示
END SUB
```

【結 果】

```
Quick BASIC
```

> **【例題 74】 数値を引数として，サブルーチンの実行**
>
> a を 88，b を 22，c と d を 0 としてサブルーチン sc を実行し，サブルーチンで a と b の積と商を求めて，メインプログラムへもどり結果を表示するプログラムをつくれ．

【解 説】 ◎ 数値を引数としてサブルーチンを実行します．

① DECLARE SUB sc(a,b,c,d)はサブルーチン sc を宣言します．このとき，引数として a，b，c，d を持ち，その値をサブルーチンへ渡します．a，b，c，d は引数が 4 個あり，各々が実数型であることを示します．ここでは，数と型が意味を持ちます．名前は意味を持ちません．サブルーチンに引数を渡せるということは従来の BASIC にないすばらしい特長です．そのよさについては項⑥で説明します．

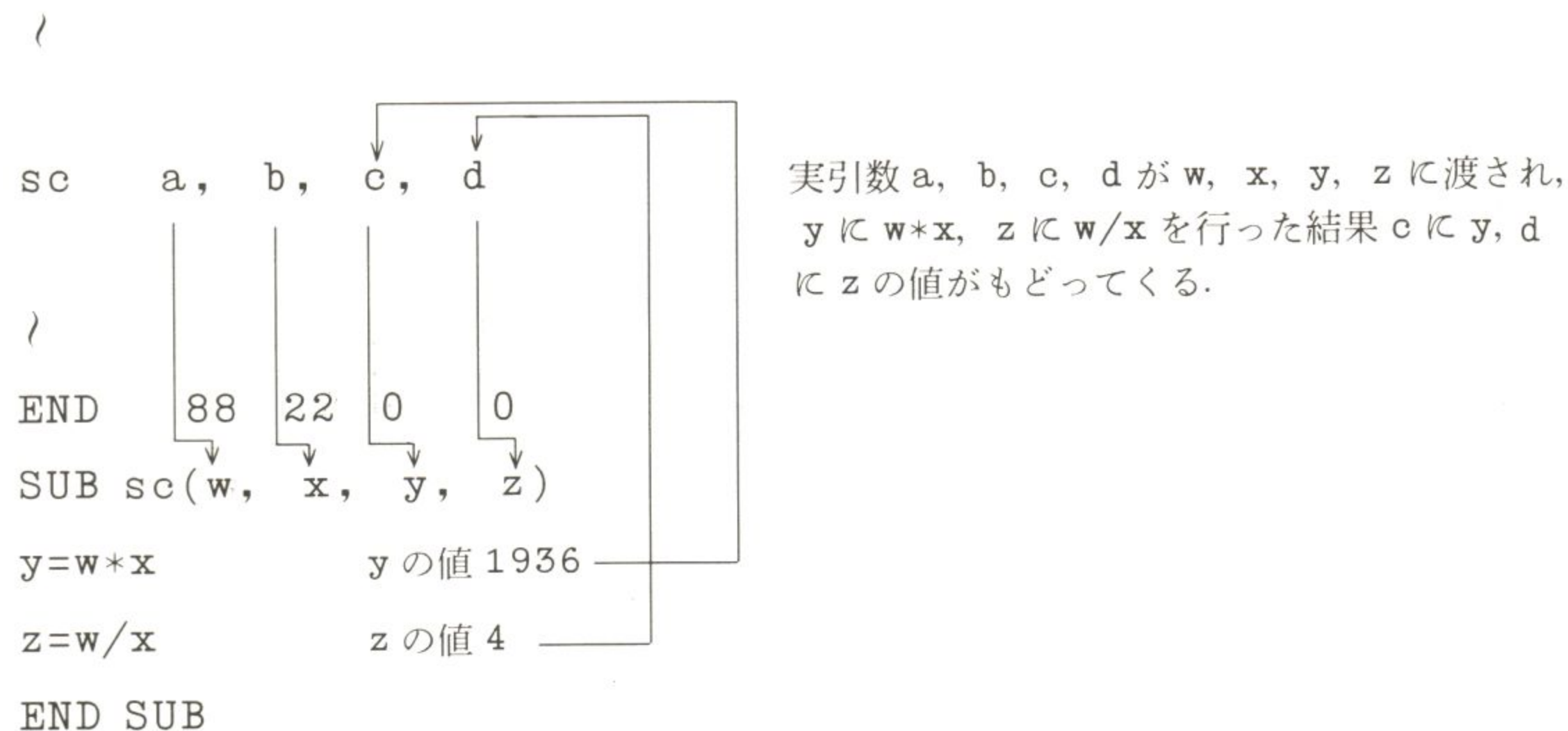
② サブルーチンでは SUB sc(w,x,y,z)とします．引数を w，x，y，z の順に受けとります．この w，x，y，z を仮引数といいます．仮引数の数と型，この場合は数が 4 個で実数型がサブルーチンを呼び出すときの引数すなわち実引数の数と型に合っていなければなりません．

③　sc a, b, c, d によってサブルーチン sc を呼び出し実行します．このとき a=88, b=22, c=0, d=0 が代入されていますから，サブルーチンの仮引数 w には a の値 88, x には b の値 22, y には c の値 0, z には d の値 0 が引き渡されます．このときの a, b, c, d を実引数といいます．

④　サブルーチン sc では a, b, c, d の値を w, x, y, z に受けて y に w*x, z に w/x の値を求めます．この結果，実引数の c は y の値 1936, d は z の値 4 となります．

⑤　この関係を図示すると次のようになります．

```
a=88, b=22, c=0, d=0 とする.
 ≀
sc    a, b, c, d
 ≀
END   88  22  0   0
SUB sc(w, x, y, z)
y=w*x          y の値 1936
z=w/x          z の値 4
END SUB
```

実引数 a, b, c, d が w, x, y, z に渡され，y に w*x, z に w/x を行った結果 c に y, d に z の値がもどってくる．

⑥　では引数をサブルーチンに渡せるメリットは何でしょうか．結論はサブルーチンの独立性が高いということです．すなわち，実引数の値によって，サブルーチンの変数の値が決まります．ですから実引数を決めずにサブルーチンへは入れません．一方，従来の BASIC では変数はどこで使っても共通ですから，本例では w, x, y, z をどこかで使っていたりすると思わぬことがおこります．

```
t=0
x=5
wa t,x
 ≀
END
SUB wa(p,q)          t=0 を p に, x=5 を q に渡す
FOR i=1  TO q        p に 1+2+…q の合計を求める
p=p+i
NEXT i
END SUB
```

p と q の値はメインプログラムから独立している．p と q がメインプログラムで使われてもそれはサブルーチンの p と q とは別のものである．

（従来の BASIC）

```
T=0
X=5
```

```
GO SUB ○○○
≀

○○○ FOR I=1  TO X      T=0,X=5 として
    T=T+I              T に 1+2+…X を求める
    NEXT I
    RETURN
```

メインプログラムのT, Xはサブルーチンの T, X と共通.
メインプログラムでT, Xを使うとサブルーチンを使わなくてもサブルーチンのT, Xの値も代入されている.

【プログラム例】

```
DECLARE SUB sc (a, b, c, d) ――――サブルーチン sc の宣言
a = 88: b = 22: c = 0: d = 0 ――――a, b, c, d に初期値を代入
sc a, b, c, d ――――サブルーチン sc の呼び出し
PRINT a; "*"; b; "="; c ――――結果の表示
PRINT a; "/"; b; "="; d
END

SUB sc (w, x, y, z) ――――サブルーチン sc
    y = w * x ――――y と z に計算結果を代入
    z = w / x
END SUB
```

【結　果】

```
88 * 22 = 1936
88 / 22 = 4
```

【例題 75】　文字列を引数とするサブルーチン

メインプログラムはサブルーチン sa1 と sa2 の宣言と各々の呼び出しとし，サブルーチン sa1 で"Let's study"，サブルーチン sa2 で"□Quick BASIC."を表示するプログラムをつくれ．ただし，サブルーチン sa1 の呼び出しで引数として"Let's study"，サブルーチン sa2 の呼び出しで引数として"□Quick BASIC." を持つものとする．

【解　説】　◎　文字列を引数としてサブルーチンを呼び出し実行します.

① DECLARE SUB sa1(a$)ではサブルーチン sa1 を宣言します．このとき(a$)によって引数が文字型であることを明示します.

② sa1"Let's study"では実引数として"Let's study"をサブルーチンへ渡します.

③ サブルーチン sa1 は SUB sa(x$)で始まります．仮引数が x$です．実引数の文字列を x$に受けとります.

```
                    ←──── a$が文字型を示す
DECLARE SUB sa1(a$)      サブルーチンの宣言
～                  ←──── 実引数は""で囲んで示す
sa1  "○○○○○"      サブルーチンの呼び出し
～              引き渡す
END            ←──── 仮引数は実引数と同じ数と型にする
SUB sa1(x$) ──── サブルーチン
～
END SUB
```

【プログラム例】

```
DECLARE SUB sa1 (a$)──────サブルーチン sa1 の宣言
DECLARE SUB sa2 (b$)──────サブルーチン sa2 の宣言
sa1 "Let's study"─────────サブルーチン sa1 の呼び出し
sa2 " Quick BASIC."───────サブルーチン sa2 の呼び出し
END

SUB sa1 (x$)──────────┬──サブルーチン sa1
    PRINT x$; ────────┼──x$の表示
END SUB ──────────────┘

SUB sa2 (y$)──────────┬──サブルーチン sa2
    PRINT y$ ─────────┼──y$の表示
END SUB───────────────┘
```

【結　果】

```
Let's study Quick BASIC.
```

【例題 76】　CALL によるサブルーチンの呼び出し

x%を 12，y%を 25，z$を"="としてサブルーチン abc を CALL で呼び出し，x%と y%の積を求め式を含めて結果を表示するプログラムをつくれ．

【解　説】　◎　CALL でサブルーチンを呼び出します．

① CALL サブルーチン（引数）によって，引数を持ってサブルーチンを呼び出します．CALL を使う場合はサブルーチンの宣言はしていなくてもサブルーチンを呼び出せます．

【プログラム例】

```
DECLARE SUB abc (x%, y%, z$)──────サブルーチン abc の宣言
x% = 12: y% = 25: z$ = "="────────x, y, z$の代入
CALL abc(x%, y%, z$) ─────────────サブルーチンの呼び出し
END

SUB abc (p%, q%, r$) ─────────────┬──サブルーチン
    PRINT p%; "*"; q%; r$; p% * q% ┼──積を求めて表示
END SUB ──────────────────────────┘
```

【結　果】

```
12 * 25 = 300
```

【例題 77】　CALL によるサブルーチンの呼び出しと値の受け渡し

【例題 76】と同様に x%を 12，y%を 25，z$を=としてサブルーチン abc を call で呼び出し，x%と y%の積を求め，式を含めて結果を表示した後，x%²を求めて，メインルーチンへもどり結果を表示するプログラムをつくれ．

【解　説】　◎　サブルーチンからメインルーチンへの値のもどし方を学びます．

① x%を 12，y%を 25，z$を"="としてサブルーチン abc を call して，x%*y%を求めて表示する部分は【例題 76】と同じです．ただし，それに加え zz%の引数をつけます．

② 次にサブルーチンの中で次のようにします．

d%=aa%*aa%

この結果 aa%は引き渡された x%の値，bb%は引き渡された y%の値，d%は aa%すなわち x%の 2 乗の値，cc$は"="となります．この状態でメインへもどります．結果はサブルーチンを call した対応する引数へ返されますから次のようになります．

```
x%      y%      z$      zz%
↑       ↑       ↑       ↑       したがって，
aa%     bb%     cc$     d%      zz%は d%すなわち aa%²すなわち x%²となる
```

【プログラム例】

```
DECLARE SUB abc (a%, b%, c$, d%) ――――― サブルーチン abc の宣言
x% = 12 ―――――――――――――――――――――――――┬ 値の代入
y% = 25                               │
z$ = "=" ―――――――――――――――――――――――――┘
CALL abc(x%, y%, z$, zz%) ――――――――― サブルーチンの呼び出し
PRINT x%; "^2="; zz% ――――――――――――― 結果の表示　zz%はサブルーチンで求める
END

SUB abc (aa%, bb%, cc$, d%) ―――――――┬ サブルーチン
    PRINT aa%; "*"; bb%; cc$; aa% * bb%―┼ x%*y%を表示
    d% = aa% * aa% ―――――――――――――――┼ x%の 2 乗を求めて d%へ代入
END SUB ―――――――――――――――――――――――┘ この値がメインルーチンへ返される
```

【結　果】

```
12 * 25 = 300
12 ^2= 144
```

【例題 78】　配列を引数とするサブルーチン

下の 30 データを配列に読み，その配列を実引数としてサブルーチン abcd を呼び出し実行して和を求めるプログラムをつくれ．

データ；2，3，5，8，4，6，1，3，8，4
　　　　2，5，6，4，2，8，9，7，2，5
　　　　3，4，4，2，7，1，9，8，8，2

【解　説】　◎　配列を実引数としてサブルーチンを実行します．

① DECLARE SUB abcd(q())で配列を引数とするサブルーチン abcd を宣言します．q()は 1 つの実数型配列が引数であることを示します．ここでは配列の数と型が問題となります．

② CALL abcd(x())は配列 x()を実引数としてサブルーチン abcd を呼び出し実行します．

③ SUB abcd(y())は y()が仮引数で実引数 x()を仮引数 y()が受けとります．したがって，y(0)は x(0)，y(1)は x(1)，y(2)は x(2)…y(29)は x(29)となります．

【プログラム例】

```
DECLARE SUB abcd (q())――――サブルーチン abcd の宣言
DIM x(30) ――――――――――┐配列 x(30)に値を代入
FOR i = 0 TO 29          │
   READ x(i)             │
NEXT i ――――――――――――――┘
CALL abcd(x())――――――――サブルーチン abcd を呼び出し実行
END
DATA 2,3,5,8,4,6,1,3,8,4
DATA 2,5,6,4,2,8,9,7,2,5
DATA 3,4,4,2,7,1,9,8,8,2

SUB abcd (y()) ――――――――┐サブルーチン abcd
   wa = 0 ――――――――――┐  │和を求める
   FOR i = 0 TO 29     │  │
      wa = wa + y(i)   │  │
   NEXT i ――――――――――┘  │
   PRINT wa ―――――――――――――┤和の表示
END SUB ―――――――――――――――┘
```

【結　果】

142

【例題 79】　文字配列を引数とするサブルーチン

下のデータを配列に読み，それを実引数としてサブルーチン abcd を呼び出し，表示するプログラムをつくれ．

データ；Tokyo，Osaka，Nagoya，Kochi，Miyazaki，
　　　　Sapporo，Sendai，Morioka，Ube，Kokura

【解　説】 ◎ 文字配列を引数としてサブルーチンを呼び出します.

① DECLARE SUB abcd(x$())によってサブルーチン abcd を宣言します. 引数は文字配列とします. x$()が文字配列を示します.

② CALL abcd(e$())で実引数を e$()としてサブルーチン abcd を呼び出します.

③ サブルーチン abcd は仮引数 y$()で実引数 e$()を受けとります.

【プログラム例】

```
DECLARE SUB abcd (x$())――――――――サブルーチン abcd の宣言
DIM e$(10) ―――――――――――――――┐配列 e$()にデータを読み込む
FOR i = 0 TO 9                    │
    READ e$(i)                    │
NEXT i ―――――――――――――――――――┘
CALL abcd(e$())―――――――――――――サブルーチン abcd を呼び出す
END
DATA Tokyo,Osaka,Nagoya,Kochi,Miyazaki
DATA Sapporo,Sendai,Morioka,Ube,Kokura

SUB abcd (y$()) ――――――――――――┬サブルーチン abcd
    FOR i = 0 TO 9 ―――――――――┐├表示
        PRINT y$(i)           ││
    NEXT i ―――――――――――――――┘│
END SUB ――――――――――――――――――――┘
```

【結　果】

```
Tokyo
Osaka
Nagoya
Kochi
Miyazaki
Sapporo
Sendai
Morioka
Ube
Kokura
```

【例題 80】 引数の型宣言の仕方

整数型, 文字型, 実数型を引数とするサブルーチン abcd を宣言し, a%を 1000, b$を"プログラム", c!を 2.56 としてサブルーチン abcd を呼び出し, 順に表示するプログラムをつくれ.

【解　説】 ◎ 宣言文の中で引数の型の書き方を示します.

① DECLARE SUB abcd(x AS INTEGER,y AS STRING,z AS SINGLE)としてサブルーチン abcd を宣言します. このとき, 引数は数と型が問題ですから x AS INTEGER によって整数型が 1 つ, y AS STRING によって文字型が 1 つ, z AS SINGLE によって実数型が 1 つを指定します. これは(x%, y$, z!)で表わすこともできます. 次に各型の表わし方を示します.

型	型の指定の仕方①	型の指定の仕方②
整数型	x%	x AS INTEGER

長整数型	x&	x AS LONG
単精度実数型	x!	x AS SINGLE
倍精度実数型	x#	x AS DOUBLE
文字型	x$	x AS STRING

【プログラム例】

```
DECLARE SUB abcd (x AS INTEGER, y AS STRING, z AS SINGLE) ── サブルーチンの宣言
a% = 1000: b$ = "ﾌﾟﾛｸﾞﾗﾑ": c! = 2.56 ─────────── 変数に初期値を代入
CALL abcd(a%, b$, c!) ─────────────────── サブルーチン abcd の呼び出し
END

SUB abcd (o%, p$, q!) ──────────────┐ サブルーチン abcd
    PRINT o%; p$; q! ───────────────┤ 表示
END SUB ────────────────────────────┘
```

【結　果】

```
1000 ﾌﾟﾛｸﾞﾗﾑ 2.56
```

【例題 81】　サブルーチンからサブルーチンの呼び出し

メインプログラムで"Quick"を表示し，サブルーチン sa を呼び出して"□BASIC"を表示した後，このサブルーチンからサブルーチン saa を引数を"□オ□マナビマショウ"として呼び出し表示するプログラムをつくれ．

【解　説】　◎　サブルーチンからサブルーチンを呼び出します．

① メインプログラムからサブルーチンを呼び出すのと同じ方法でサブルーチンから別のサブルーチンを呼び出すことができます．

② 本例ではメインプログラムで"Quick"を表示した後，引数なしでサブルーチン sa を呼び出し，サブルーチン sa で"□BASIC"を表示し，次に，そこからサブルーチン saa を引数を"□オ□マナビマショウ"として呼び出し表示します．

【プログラム例】

```
DECLARE SUB sa () ────────── サブルーチン sa の宣言
DECLARE SUB saa (a$) ─────── サブルーチン saa の宣言
PRINT "Quick"; ───────────── "Quick"の表示
sa ───────────────────────── サブルーチン sa の呼び出し
END

SUB sa ──────────────────┐ サブルーチン sa
    PRINT " BASIC"; ─────┤ "□BASIC"の表示
    saa " ｵ ﾏﾅﾋﾞ ﾏｼｮｳ" ──┤ サブルーチン saa の呼び出し
END SUB ─────────────────┘

SUB saa (x$) ────────────┐ サブルーチン saa
    PRINT x$ ────────────┤ x$の表示
END SUB ─────────────────┘
```

【結　果】

```
Quick BASIC オ マナビマショウ
```

〔演習問題〕

(87)　"プロシージャ　ノ　ヨビダシ"を表示した後サブルーチン a を呼び出し，サブルーチン a と表示するプログラムをつくれ.

(88)　a を 50，b を 36，c と d を 0 としてそれを実引数としてサブルーチン sb を呼び出し実行して，a と b の和と差を求め，メインプログラムにもどって結果を表示するプログラムをつくれ.

(89)　"Quick BASIC"を表示した後サブルーチン a を呼び出して，"□MS-DOS"を表示するプログラムをつくれ.

(90)　サブルーチン abcd を引数が 2 つの実数型として宣言し，125.357 と 18.2546 を実引数としてサブルーチン abcd を呼び出し，積を求めて表示するプログラムをつくれ.

(91)　下のデータを使って，配列 e$(10) と f(10) に読み，これを実引数としてサブルーチン abcd を呼び出し，表示するプログラムをつくれ.

データ；Tokyo，10，Osaka，8，Nagoya，6，Kochi，
3，Miyazaki，4，Sapporo，5，Sendai，4，
Morioka，3，Ube，2，Kokura，4

(92)　メインプログラムで aa$を"□BASIC"，"bb$を"□C"，cc$を"□Fortran"として，サブルーチン a，b，c を呼び出し，"Quick"の後に各々aa$，bb$，cc$の内容を追加して表示するプログラムをつくれ.

(93)　次の配列データをサブルーチンに渡し，番号を入力して，その順番のデータを表示するプログラムをつくれ.

Ikebukuro，Otsuka，Sugamo，Komagome，Tabata，
Nishinippori，Nippori，Uguisudani，Ueno，
Okachimachi，Akihabara，Kanda，Tokyo

(94)　(93) の配列データをサブルーチンに引き渡し，"T"で始まる駅名を表示するプログラムをつくれ.

(95)　3.14159 を実引数としてサブルーチン a にわたし，半径 10 の円の面積を求めるプログラムをつくれ.

(96)　実数 23.857 と整数 3 を実引数としてサブルーチン a を呼び出し，23.857^3 を求めるプログラムをつくれ.

(97)　整数 x%，y%を入力して，x%の y%乗をサブルーチンで求めて表示するプログラムをつくれ. ただし，x%の y%乗が 10000 をこえるときはさらにサブルーチンへ分岐して"x%^y%>=10000"と表示するものとする.

〔解 答〕

(87) **【プログラム例】**

```
DECLARE SUB a ()―――――――――サブルーチン a の宣言
PRINT "ﾌﾟﾛｼｰｼﾞｬ ﾉ ﾖﾋﾞﾀﾞｼ"――――表示
a ―――――――――――――――――――サブルーチン a の呼び出し
END

SUB a ―――――――――――――――――サブルーチン a
    PRINT "ｻﾌﾞﾙｰﾁﾝ a"―――――┐表示
END SUB――――――――――――――――┘
```

【結　果】

```
ﾌﾟﾛｼｰｼﾞｬ ﾉ ﾖﾋﾞﾀﾞｼ
ｻﾌﾞﾙｰﾁﾝ a
```

(88) **【プログラム例】**

```
DECLARE SUB sb (a, b, c, d)――――サブルーチン sb の宣言
a = 50: b = 36: c = d = 0 ―――――a, b, c, d の初期値を代入
sb a, b, c, d ―――――――――――――サブルーチン sb の呼び出し
PRINT a; "+"; b; "="; c ―――――┐結果の表示
PRINT a; "-"; b; "="; d ―――――┘
END

SUB sb (w, x, y, z)――――――――――サブルーチン sb
    y = w + x ――――――――――┐┐和と差を求める
    z = w - x ――――――――――┘│
END SUB ――――――――――――――――――┘
```

【結　果】

```
50 + 36 = 86
50 - 36 = 14
```

(89) **【プログラム例】**

```
DECLARE SUB a ()――――――――サブルーチン a の宣言
PRINT "Quick Basic";――――表示
CALL a ――――――――――――――サブルーチン a の呼び出し
END

SUB a ――――――――――――――――サブルーチン a
    PRINT " MS-DOS"――――┐表示
END SUB ―――――――――――――┘
```

【結　果】

```
Quick Basic MS-DOS
```

(90)　**【プログラム例】**

```
DECLARE SUB abcd (x AS SINGLE, y AS SINGLE)———サブルーチン abcd の宣言
a! = 125.357: b! = 18.2546 ——————————a と b に値を代入
CALL abcd(a!, b!)—————————————————サブルーチン abcd の呼び出し
END

SUB abcd (o!, p!)—————————————————サブルーチン abcd
    PRINT o!; "*"; p!; "="; o! * p!———————┬積を表示
END SUB———————————————————————————————┘
```

【結　果】

```
125.357 * 18.2546 = 2288.342
```

(91)　**【プログラム例】**

```
DECLARE SUB abcd (x$(), y()) ——————————————サブルーチンの宣言
DIM e$(10), f(10) ————————————————————————配列にデータの読み込み
FOR i = 0 TO 9
    READ e$(i), f(i)
NEXT i ———————————————————————————————┬サブルーチン abcd の呼び出し
CALL abcd(e$(), f()) ————————————————┘
END
DATA Tokyo,10,Osaka,8,Nagoya,6,Kochi,3,Miyazaki,4
DATA Sapporo,5,Sendai,4,Morioka,3,Ube,2,Kokura,4

SUB abcd (y$(), z()) ——————————————————┬サブルーチン abcd
    FOR i = 0 TO 9 ———————————————┬————表示
        PRINT y$(i ); z(i)          │   │
    NEXT i ———————————————————————┘   │
END SUB ———————————————————————————————┘
```

【結　果】

```
Tokyo 10
Osaka 8
Nagoya 6
Kochi 3
Miyazaki 4
Sapporo 5
Sendai 4
Morioka 3
Ube 2
Kokura 4
```

(92)　**【プログラム例】**

```
DECLARE SUB a (z$)———————┬サブルーチン a, b, c の宣言
DECLARE SUB b (z$)       │
DECLARE SUB c (z$)———————┘
aa$ = " Basic"———————————┬値の代入
bb$ = " C"               │
cc$ = " Fortran"—————————┘
```

```
CALL a(aa$)─────────────┐─サブルーチンの呼び出し
CALL b(bb$)             │
CALL c(cc$) ────────────┘
END

SUB a (zz$)─────────────┐─サブルーチン a
    PRINT "Quick" + zz$ │
END SUB ────────────────┘

SUB b (zz$) ────────────┐─サブルーチン b
    PRINT "Quick" + zz$ │
END SUB ────────────────┘

SUB c (zz$) ────────────┐─サブルーチン c
    PRINT "Quick" + zz$ │
END SUB ────────────────┘
```

【結　果】

```
Quick Basic
Quick C
Quick Fortran
```

(93)　**【プログラム例】**

```
DECLARE SUB a (x$())────────サブルーチン a の宣言
DIM z$(1 TO 13) ─────────┐─配列 z$に値を読み込む
FOR i = 1 TO 13          │
    READ z$(i)           │
NEXT i───────────────────┘
CALL a(z$()) ───────────────配列の値を実引数としてサブルーチン a を呼び出す
END
DATA Ikebukuro,Otsuka,Sugamo,Komagome,Tabata
DATA Nishinippori,Nippori,Uguisudani,Ueno
DATA Okachimachi,Akihabara,Kanda,Tokyo

SUB a (zz$()) ───────────┐─サブルーチン a
    INPUT x%             │ x%を入力して添字を x%とする配列の値を表示
    PRINT zz$(x%)────────┘
END SUB
```

【結　果】

```
? 3                3 を入力の例
Sugamo
? 7                7 を入力の例
Nippori
```

(94)　**【プログラム例】**

```
DECLARE SUB a (x$()) ────────────────サブルーチン a の宣言
DIM z$(1 TO 13)─────────────────────┐─配列 z$に値を読み込む
FOR i = 1 TO 13                     │
    READ z$(i)                      │
NEXT i ─────────────────────────────┘
```

```
CALL a(z$()) ──────────── 配列の値を実引数としてサブルーチン a を呼び出す
END
DATA Ikebukuro,Otsuka,Sugamo,Komagome,Tabata
DATA Nishinippori,Nippori,Uguisudani,Ueno
DATA Okachimachi,Akihabara,Kanda,Tokyo

SUB a (zz$()) ─────────────── サブルーチン a
    FOR i = 1 TO 13 ─────────── "T"で始まる文字列を表示
        IF ASC(zz$(i)) = 84 THEN PRINT zz$(i)
    NEXT i
END SUB
```

【結　果】

```
Tabata
Tokyo
```

(95)【プログラム例】

```
DECLARE SUB a (x!) ───────── サブルーチン a を宣言
z! = 3.14159 ─────────────── z!に値を代入
CALL a(z!) ───────────────── z!を実引数としてサブルーチン a を呼び出す
END


SUB a (zz!) ──────────────── サブルーチン a
    s! = 10 * 10 * zz! ───── s!に半径10の円の面積を求めて表示
    PRINT s!
END SUB
```

【結　果】

```
314.159
```

(96)【プログラム例】

```
DECLARE SUB a (x!, y%) ───── サブルーチン a の宣言
z! = 23.857 ──────────────── z!, zz%に値を代入
zz% = 3
CALL a(z!, zz%) ──────────── サブルーチン a を呼び出す
END


SUB a (qq!, pp%) ─────────── サブルーチン a
    PRINT qq! ^ pp% ──────── z!の zz%累乗を求めて表示
END SUB
```

【結　果】

```
13578.37
```

(97) **【プログラム例】**

```
DECLARE SUB b (r%, s%) ──────────┬─サブルーチン a, b の宣言
DECLARE SUB a (xx%, yy%)─────────┘
INPUT x%, y% ────────────────────┬─x%, y%の入力
CALL a(x%, y%) ──────────────────┘ サブルーチン a を呼び出す
END

SUB a (qq%, pp%) ────────────────┬─サブルーチン a
    IF qq% ^ pp% <= 10000 THEN   │ x%, y%が 10000 以下のとき表示,
        PRINT qq% ^ pp%          │ こえるときサブルーチン b を呼び出す
    ELSE                         │
        CALL b(qq%, pp%)         │
    END IF                       │
END SUB ─────────────────────────┘

SUB b (r%, s%) ──────────────────┬─サブルーチン b
    PRINT r%; "^"; s%; ">=10000" │ 10000 以上の表示
END SUB ─────────────────────────┘
```

【結　果】

```
? 23,5                        23, 5 を入力の例
 23 ^ 5 >=10000
? 3,2                         3, 2 を入力の例
 9
? 3,5                         3, 5 を入力の例
 243
```

12章　プロシジュア・ファンクション

DECLARE FUNCTION, FUNCTION, END FUNCTION

【例題82】　プロシジュア・ファンクション

数xを入力してxの階乗(x!)（x!は単精度実数型を示すのではなくxの階乗を示す数学記号である）を求めるプログラムをつくれ．ただし，階乗を求める部分にFUNCTIONを使うものとする．

【解　説】　◎　FUNCTIONを使って階乗を求めます．

① FUNCTIONはサブルーチン同様の考えで使います．SUBとの違いはFUNCTIONでは自身が値を持つことです．具体的にみていきましょう．

② DECLARE FUNCTION Kaijo(y AS DOUBLE)ではFUNCTIONとしてKaijoを宣言します．このとき引数がDOUBLE型であることを示します．これはSUBの宣言の仕方と同様です．

③ FUNCTION Kaijo(a#)～END FUNCTIONはSUB　名前（仮引数）～END SUBと同様です．Kaijoという名前のFUNCTIONを示し，仮引数としてa#を持っています．実引数をa#に受けとります．

本例の処理の内容はkaijoに階乗の結果を持ちます．次にa#が0のときKaijoに0，a#が0でないときKaijoにj#を代入します．すなわち，Kaijo自身が値を持つことがSUBと違います．SUBでは引数どうしで値をやりとりしましたが，SUBの名前は値を持っていませんでした．

④ メインプログラムのKaijo(x#)によりx#を引数として，FUNCTION Kaijoを呼び出し実行します．FUNCTIONでは階乗を求めてKaijoに代入していますのでKaijo(x#)はx!（xの階乗）となります．SUBと違ってKaijo(x#)自身が値を持っています．

【プログラム例】

```
DECLARE FUNCTION kaijo (y AS DOUBLE)――――――ファンクション kaijo の宣言
INPUT x# ―――――――――――――――――――――――――x#の入力
PRINT x#; "!"; "="; kaijo(x#) ――――――――――kaijo を呼び出し実行し結果を表示
END

FUNCTION kaijo (a#) ―――――――――――――――┐―ファンクション kaijo
    j# = 1 ―――――――――――――――――――――┐ |―階乗を j#に求める
    FOR i = a# TO 2 STEP -1              | |
        j# = j# * i                      | |
    NEXT i ―――――――――――――――――――――┘ |
    IF a# = 0 THEN kaijo = 0 ELSE kaijo = j# ―|―ファンクション kaijo に階乗の結果を代入
END FUNCTION ―――――――――――――――――――――┘
```

【結　果】

```
? 11                          11 を入力の例
 11 != 3.99168E+07

? 9                           9 を入力の例
 9 != 362880

? 4                           4 を入力の例
 4 != 24
```

【例題 83】　FUNCTION に引数を 2 つ渡す

数 x, y を入力して，次にその値をファンクション seki に与えて，積を求め，結果をメインプログラムに返して表示するプログラムをつくれ.

【解　説】　◎　FUNCTION に引数を 2 つ渡します.

① DECLARE FUNCTION seki(x,y)で，ファンクション seki を宣言します．引数は x と y です．これは浮動小数点型の 2 つの値です．ここでは変数名 x，y が問題ではなく，数（2 個）と型が問題です.

② ファンクション seki は次のとおりです.

```
FUNCTION seki(a,b)
seki=a*b
END FUNCTION
```

名前は seki です．引数は a，b で，xx の値を a，yy の値を b に引きうけます．a*b の結果がファンクション seki の値となります.

③ ファンクションの値は 3 行目の seki(xx,yy)で表示します．xx，yy を実引数としてファンクション seki を実行し，その値が seki(xx,yy)です.

【プログラム例】

```
DECLARE FUNCTION seki (x, y) ――――――― ファンクション seki の宣言
INPUT xx, yy ―――――――――――――――――――― xx，yy の入力
PRINT xx; "*"; yy; "="; seki(xx, yy)―― ファンクション seki(xx,yy)の表示
END

FUNCTION seki (a, b) ―――――――――――┐ ファンクション seki
    seki = a * b                     │
END FUNCTION ―――――――――――――――――――┘
```

【結　果】

```
? 15,3                        15 と 3 を入力の例
 15 * 3 = 45

? -25,9                       -25 と 9 を入力の例
-25 * 9 =-225
```

【例題84】　ファンクションの値でループを終了

文字を入力し，"E"または"e"が入力されたとき全体を順につないで表示するプログラムをつくれ．ただし，"E"または"e"が入力されたときファンクション handan の値を 1 として文字の入力を終了し，それ以外では handan の値を 0 として，次の入力を行うものとする．

【解　説】　◎　WHILE　条件の条件をファンクションの値で行います．

①　次のループはファンクション handan の値が 1 でないときくり返しをします．1 のときくり返しを終了して s$を表示します．

```
WHILE handan(b$)<>1   ―― ファンクション handan を実行して 1 以外なら
                          くり返し，1 ならループの外へ
      s$=s$+b$        ┐
WEND                  └ 入力された b$を順に s$につないでいく
PRINT s$
```

②　ファンクション handan は文字列 b$を入力して"E"または"e"のとき handan の値を 1 とし，それ以外では 0 とします．引数 b$はそのままメインプログラムの b$に引き渡されます．

【プログラム例】

```
DECLARE FUNCTION handan (a AS STRING)―― ファンクション handan の宣言引数は文字列が 1 つ
s$ = ""――――――――――――――――――――――┐ handan が 1 以外のときくり返し s$に文字をつないでいく
WHILE handan(b$) <> 1                   │
     s$ = s$ + b$                       │
WEND ―――――――――――――――――――――――――――┘
PRINT s$ ――――――――――――――――――――――― s$の表示
END

FUNCTION handan (b$) ――――――――――――― ファンクション handan
    INPUT b$                                b$の入力
    IF b$ = "E" OR b$ = "e" THEN z = 1 ELSE z = 0   b$が"E"または"e"のときファンクション
    handan = z                              の値が 1 その他は 0
END FUNCTION
```

【結　果】

```
? b        b, o, o, k, e を入力の例
? o
? o
? k
? e
book
```

【例題 85】　配列の値をファンクションへ引き渡す

下のデータを元にして，数 x を入力し，最初から x 個の平均を求めて表示するプログラムをつくれ．ただし，メインプログラムで x 個の入力をし，配列に元データをよんで，x 個とその配列の値をファンクションに引き渡し，ファンクションで平均を求めるものとする．

5，3，4，8，6，7，1，9，5，6
8，9，2，4，5，3，2，9，8，4
9，7，5，8，4，2，2，9，2，5

【解　説】　◎　配列をファンクションへ渡します．

① 配列名をかくことで配列をファンクションへ引き渡すことができます．

【プログラム例】

```
DECLARE FUNCTION heikin (x(), y) ――― ファンクション heikin の宣言，配列と整数型を引数とする
DIM a(30) ―――――――――――――――――――― 配列の宣言
INPUT xx ――――――――――――――――――――― xx%の入力
FOR i = 1 TO xx ―――――――――――――――┐ 配列 a%に xx 個のデータを読む
    READ a(i)                       │
NEXT i ―――――――――――――――――――――――┘
h = heikin(a(), xx) ―――――――――――――┐ ファンクション heikin を表示
PRINT xx; "コ ノ ヘイキンヘン="; h ――――┘
END
DATA 5,3,4,8,6,7,1,9,5,6
DATA 8,9,2,4,5,3,2,9,8,4
DATA 9,7,5,8,4,2,2,9,2,5

FUNCTION heikin (n(), p) ――――――――― ファンクション heikin，配列を n%(i)に個数を p%にうける
    t = 0 ―――――――――――――――――――――┐ 和を求める
    FOR i = 1 TO p                   │
        t = t + n(i)                 │
    NEXT i ―――――――――――――――――――――┘
    heikin = t / p ――――――――――――――― 平均を求める，平均を heikin の値とする
END FUNCTION
```

【結　果】

```
? 10                              10 を入力の例
 10 コ ノ ヘイキンヘン= 5.4

? 16                              16 を入力の例
 16 コ ノ ヘイキンヘン= 5.3125

? 30                              30 を入力の例
 30 コ ノ ヘイキンヘン= 5.366667

? 1                               1 を入力の例
 1 コ ノ ヘイキンヘン= 5
```

〔演習問題〕

(98)　数 xx，yy を入力し，これをファンクション wa に引き渡して和を求め結果をメインプログラムにもどして表示するプログラムをつくれ．

(99)　文字列 x$を入力し，その文字列をファンクション mojisu に渡しその文字列の文字数をファンクション mojisu の値としてメインプログラムにもどして表示するプログラムをつくれ．

(100)　整数を入力して，その値が 0 のとき値を 999 とし，それ以外では値を 0 とするファンクション handan をつくり，引数として，入力された値をメインプログラムにもどして和を求め，0 が入力されたとき和を表示するプログラムをつくれ．

(101)　整数 x%を入力して，$3x^2+2x+5$ の値を求めるプログラムをつくれ．ただし，式の値は FUNCTION を使って求めるものとする．

(102)　整数 x%を入力して，x%度の sin の値を求めるプログラムをつくれ．

(103)　整数 x%を入力して，1～x%の和を求めるプログラムをつくれ．

〔解　答〕

(98) **【プログラム例】**

```
DECLARE FUNCTION wa (x, y) ――――――――― ファンクション和の宣言
INPUT xx, yy ―――――――――――――――――― xx，yy の入力
PRINT xx; "+"; yy; "="; wa(xx, yy) ―― ファンクション wa の表示
END

FUNCTION wa (a, b) ―――――――――┐
    wa = a + b                  ├ ファンクション wa
END FUNCTION ――――――――――――――┘
```

【結　果】

```
? 154,854                     154 と 854 を入力の例
 154 + 854 = 1008

? 21,8956                     21 と 8956 を入力の例
 21 + 8956 = 8977
```

(99) **【プログラム例】**

```
DECLARE FUNCTION mojisu (a AS STRING)―― ファンクション mojisu の宣言
INPUT x$ ――――――――――――――――――――――― x$の入力
PRINT mojisu(x$) ―――――――――――――――― ファンクション mojisu の表示
END

FUNCTION mojisu (b$) ――――――――――┐ ファンクション mojisu
    mojisu = LEN(b$)              │ 文字列の文字数を求める
END FUNCTION ―――――――――――――――――┘
```

【結　果】

```
? moonlight          moonlightを入力の例
 9
```

(100)【プログラム例】

```
DECLARE FUNCTION handan (a AS INTEGER)――ファンクション handan の宣言
wa = 0 ―――――――――――――――――――――――――――――wa を求めるくり返し
WHILE handan(b%) <> 999                   入力された値 b%はファンクション
     wa = wa + b%                         handan から引き渡される
WEND ――――――――――――――――――――――――――――――
PRINT wa ―――――――――――――――――――――――――――和の表示
END

FUNCTION handan (b%) ―――――――――――――――――――――ファンクション handan
    INPUT b% ――――――――――――――――――――――――――――――b%の入力
    IF b% = 0 THEN handan = 999 ELSE handan = 0 ―b%が 0 のときファンクションの値を 999,
END FUNCTION ――――――――――――――――――――――――――――――その他のとき 0 とする
```

【結　果】

```
? 5              5, 7, 9, 7, 2, 4, 5, 0 を入力の例
? 7
? 9
? 7
? 2
? 4
? 5
? 0
 39
```

(101)【プログラム例】

```
DECLARE FUNCTION a (a AS INTEGER)――――a の定義
INPUT x% ――――――――――――――――――――――――x%の入力
PRINT a(x%) ―――――――――――――――――――――a を呼び出し結果を表示
END

FUNCTION a (z%) ――――――――――――――――――ファンクション a
    a = 3 * z% * z% + 2 * z% + 5     3x%²+2x%+5 を求める
END FUNCTION ―――――――――――――――――――
```

【結　果】

```
? 2          2 を入力の例
 21
? 8          8 を入力の例
 213
```

(102)【プログラム例】

```
DECLARE FUNCTION a (a AS INTEGER)――――a の定義
INPUT x% ――――――――――――――――――――――――x%の入力
PRINT a(x%) ―――――――――――――――――――――a を呼び出し結果を表示
END
```

```
FUNCTION a (z%) ──────────────┬ファンクション a
    a = SIN(z% / 180 * 3.14159)  │sin (x%度) を求める
END FUNCTION ─────────────────┘
```

【結　果】

```
? 67            67 を入力の例
 .9205045
```

(103)【プログラム例】

```
DECLARE FUNCTION a (a AS INTEGER) ────a の定義
INPUT x% ──────────────────────────x%の入力
PRINT a(x%) ───────────────────────a を呼び出し結果を表示
END

FUNCTION a (z%) ──────────────┬ファンクション a
    FOR i% = 1 TO z%            │1 から x%の和を求める
        zz% = zz% + i%          │
    NEXT i%                     │
    a = zz%                     │
END FUNCTION ─────────────────┘
```

【結　果】

```
? 5             5 を入力の例
 15
? 100           100 を入力の例
 5050
```

13章　グローバル変数とローカル変数，プログラムの連結

COMMON, COMMON SHARED, CHAIN, STATIC, SHARED

【例題86】　グローバル変数とローカル変数

変数 pi を 3.14159 と CONST で宣言・代入し，次に半径 r を入力して，円の面積（pi r^2）を求めて表示し，サブルーチン ss を引数を r として呼び出し，円周長（2pi・r）を求めて表示するプログラムをつくれ.

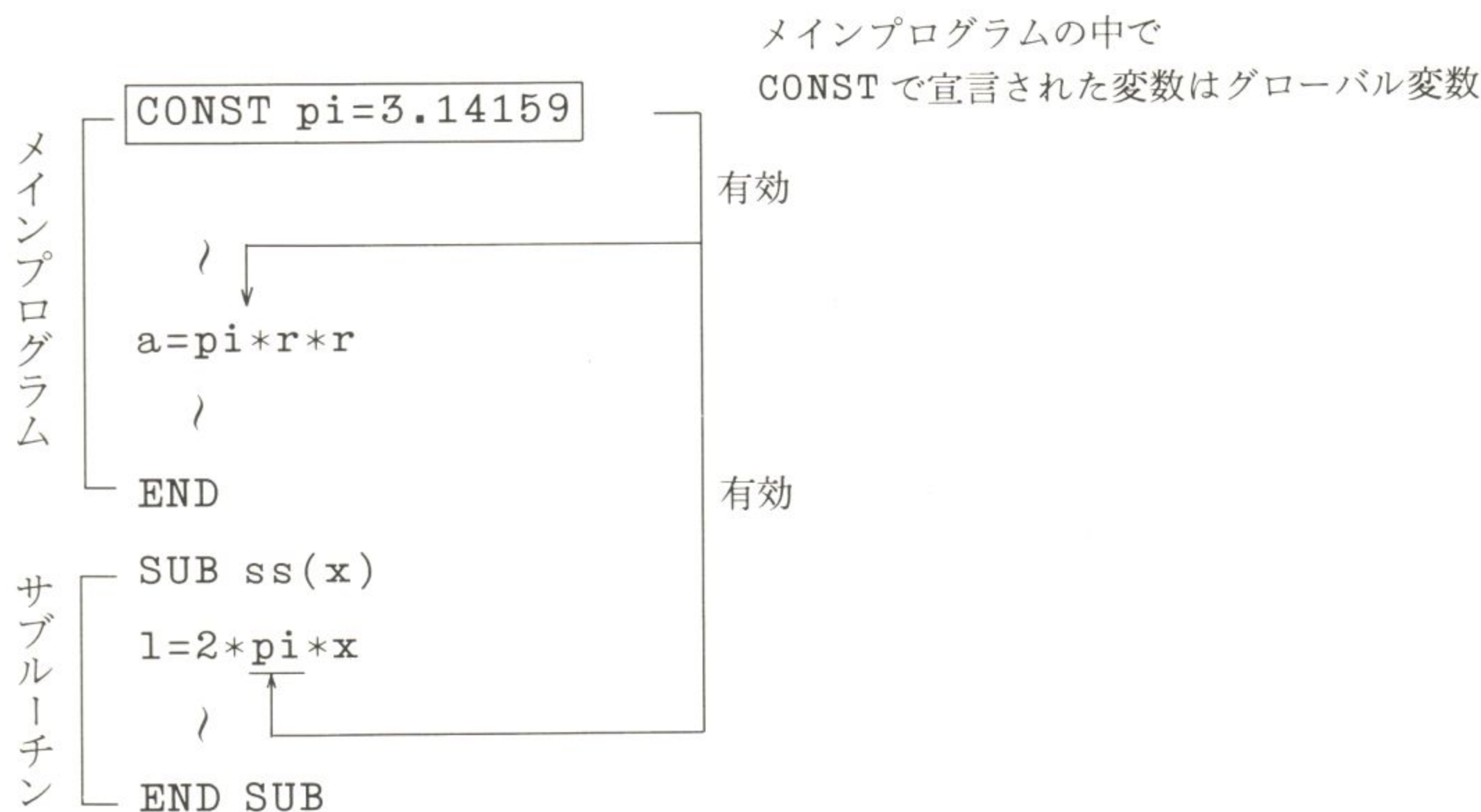
【解　説】　◎　グローバル変数の有効範囲とローカル変数の有効範囲

① Quick BASIC では変数の有効範囲によってグローバル変数とローカル変数に分けます．グローバル変数はその値がすべてのモジュールで有効な変数です．従来の BASIC では変数はプログラムの中ではすべて有効でしたから，すべての変数がグローバル変数であったともいえます．一方，特定のモジュールの中だけで有効な変数をローカル変数といいます．Quick BASIC では特に指定のない限り，変数はローカル変数です．このことはサブルーチンやファンクションの独立性が高いことを示します.

② メインプログラムの中で宣言・代入された定数 CONST pi=3.14159 はグローバルです．本例ではメインプログラムの中の pi は 3.14159 であり，サブルーチン ss(x)の中の pi も 3.14159 が有効です.

③ プロシジュアの中で宣言される CONST はローカルで，そのプロシジュアの中でのみ有効です.

メインプログラムの中で
CONST で宣言された変数はグローバル変数

```
メインプログラム
  CONST pi=3.14159        ─┐ 有効
     ~                     │
  a=pi*r*r                 │
     ~                     │
  END                      │ 有効
サブルーチン               │
  SUB ss(x)                │
  l=2*pi*x  ←──────────────┘
     ~
  END SUB
```

④ メインプログラムの中で入力された値を持つ変数 r はローカルです．したがって，メインプログラムの中だけで有効です．したがって，CALL ss(r)では r の値をサブルーチンへ引数として渡し

ます．なお，メインプログラムの変数 s もローカルです．

⑤　サブルーチンでは 2*pi*x を求めて l に代入し，x と l を表示します．pi の値はグローバルですから 3.14159 が生きています．x の値は関数の引数として r の値が引き渡されています．結果は変数 l に代入されます．l はローカル変数ですからサブルーチンの中でのみ有効です．

⑥　サブルーチンの中で x のかわりに r の値を表示させてみます．結果は 0 です．すなわちメインプログラムの r の値はローカルですからサブルーチンの中では 0 です．r の値はサブルーチンの引数として仮引数 x に渡されているのです．

【参考プログラム例】

```
SUB ss (x)
l = 2 * pi * x
PRINT r; l ———————— x の代わりに r を表示した例
END SUB
```

【結　果】

```
? 3.84                      3.84 を入力の例
 3.84   46.32463
 0   24.12741               r は 0 になっていることに注意
```

【プログラム例】

```
DECLARE SUB ss (x!) ———— サブルーチン ss の宣言
CONST pi = 3.14159 ———— 定数の宣言
INPUT r ———————————— r の入力
s = pi * r * r ———————— 円の面積の計算
PRINT r; s ——————————— 結果の表示
CALL ss(r) ——————————— サブルーチンを呼び出し，引数を r とする
END

SUB ss (x) ———————————— サブルーチン ss．仮引数は x
    l = 2 * pi * x ———————— 円周長を求める
    PRINT x; l ———————————— 半径と円周長の表示
END SUB
```

【結　果】

```
? 2.5 ———————————————— 2.5 を入力の例
 2.5   19.63494
 2.5   15.70795
```

【例題 87】　**整数型変数と長整数型変数を COMMON で宣言**

変数 a，b，c を整数型，d，e，f を長整数型で COMMON 宣言し，a，b，c を入力して，d に積，e に各々の 2 乗の積，f に d の 2 乗を求めるプログラムをつくれ．

【解　説】　◎　整数型変数と長整数型変数を COMMON で宣言します．

①　次の形で COMMON 宣言をします．COMMON で宣言された変数はグローバル変数です．

```
COMMON a,b,c AS INTEGER
```

a，b，c を整数型で宣言

COMMON d,e,f AS LONG

d，e，f を長整数型で宣言

【プログラム例】

```
COMMON a, b, c AS INTEGER ─┐ 宣言
COMMON d, e, f AS LONG ────┘
INPUT a, b, c ──────────── 入力
d = a * b * c ────────────┐ 積
e = a * a * b * b * c * c │
f = d * d ────────────────┘
PRINT a; b; c ────────────┐ 表示
PRINT d; e; f ────────────┘
END
```

【結　果】

```
? 12,10,6 ──────────── 12，10，6 を入力の例
 12   10   6
 720   518400   518400
```

【例題 88】　単精度実数型変数と倍精度実数型変数を COMMON で宣言

変数 a，b を単精度実数，c，d を倍精度実数と宣言して，a，b を入力して，積を c に，商を d に代入して a，b，c，d を表示するプログラムをつくれ．

【解　説】　◎　単精度実数型変数と倍精度実数型変数を COMMON で宣言をします．

① 次の形で COMMON 宣言します．COMMON で宣言された変数はグローバル変数です．

COMMON a,b AS SINGLE

a，b を単精度実数として宣言

COMMON c,d AS DOUBLE

c，d を倍精度実数として宣言

【プログラム例】

```
COMMON a, b AS SINGLE ─┐ 宣言
COMMON c, d AS DOUBLE ─┘
INPUT a, b ──────────── 入力
c = a * b ─────────────┐ 積と商
d = a / b ─────────────┘
PRINT a; b ────────────┐ 表示
PRINT c; d ────────────┘
END
```

【結　果】

```
? 25.325,12.7896          25.325 と 12.7896 を入力の例
 25.325   12.7896
 323.8966   1.980124473571777
```

【例題 89】　文字型変数を COMMON で宣言

変数 a を 10 文字の COMMON 宣言をして a に文字列を入力して表示するプログラムをつくれ.

【解　説】　◎　文字型変数を COMMON で宣言します.

①　次の形で文字変数の文字数を決めて COMMON 宣言をします. COMMON で宣言された変数はグローバル変数です.

```
COMMON a AS STRING*10
```

変数 a を 10 文字列として COMMON 宣言

【プログラム例】

```
COMMON a AS STRING * 10 ――― 宣言
INPUT a ――――――――――― 入力
PRINT a ――――――――――― 表示
END
```

【結　果】

```
? abcdefg                a……g ⏎ の例
abcdefg
? abcdefghij             a……j ⏎ の例
abcdefghij
? abcdefghijklmn         a……n ⏎ の例
abcdefghij
```

【例題 90】　グローバル変数

pi を 3.14159 として CONST で宣言し, 次に変数 r をグローバルに宣言してから r に半径を入力して, 円の面積を求めて表示した後, サブルーチンを使って, 2*pi*r (円周長) を求めて表示するプログラムをつくれ.

【解　説】　◎　プロシジュアで有効なグローバル変数を宣言します.

①　メインプログラムで次のように宣言された変数はグローバルです. モジュール間を通じて共通に使えます.

COMMON SHARED　変数

②　メインプログラムで r を入力し pi*r*r を求めた後サブルーチンで 2*pi*r を求めます. 今回はサブルーチンへの引数はありません. r の値は入力された値が生きていて半径 r の円周長が求まります. このように r の値はサブルーチンの中でも有効なのです.

③　【例題 86】の r の値, r をサブルーチンへ渡す方法と比べてみてください. 従来の BASIC になれた人は r が共通に使える方が便利で簡単と思われるかも知れません. しかし, プログラムが複雑になると, サブルーチンや関数の独立性が高い方が便利になってくるのです.

【プログラム例】

```
DECLARE SUB ss ()———————サブルーチン ss の宣言
CONST pi = 3.14159———————定数の宣言
COMMON SHARED r ———————変数 r をグローバルに宣言
INPUT r ————————————————r の入力
s = pi * r * r——————————円の面積の計算
PRINT r; s ——————————————半径と面積の表示
CALL ss —————————————————サブルーチンの呼び出し，引数はない.
END                      半径 r はグローバルなのでサブルーチンでも使える

SUB ss ——————————————————サブルーチン
    l = 2 * pi * r———————円周長の計算
    PRINT r; l———————————半径と円周長の表示
END SUB
```

【結　果】

```
? 12.5                    12.5 を入力の例
 12.5   490.8734
 12.5   78.53975
```

【例題 91】　ローカル変数

x を 125.33，y を 654.1 として x*y を xy に求めてからサブルーチン ss を呼び出し，x を 54.2，y を 32.58 として x*y を xy に求めて表示し，メインプログラムに戻って，最初の x*y の値を表示するプログラムをつくれ.

【解　説】　◎　ローカル変数の働きについて調べます.

① メインプログラムで x=5.33，y=654.1 として x*y を求めて xy に代入します.

② 続いてサブルーチンで x=54.2，y=32.58 として x*y を求めて xy に代入します．結果を表示します.

③ メインルーチンに戻って，最初の x と y および xy の値を表示します．x はサブルーチンで 54.2 を代入していますが元の 125.33 が生きています．y も同様に 654.1 が生きています．xy は 54.2*654.1 です．つまり，変数 x，y，xy はローカル変数のためメインプログラムの中の x，y，xy とサブルーチンの中の x，y，xy ではお互いに独立しているのです.

【プログラム例】

```
DECLARE SUB ss () ——————————サブルーチン ss の宣言
x = 125.33 ————————————┐———x と y に値を代入
y = 654.1 —————————————┘
xy = x * y —————————————————xy に x と y の積を代入
CALL ss ————————————————————サブルーチン ss を呼び出す
PRINT x; "*"; y; "="; xy ———x，y，xy の表示
END

SUB ss —————————————————————サブルーチン ss
    x = 54.2 ——————————┐———x と y に値を代入
    y = 32.58——————————┘
```

```
    xy = x * y ———————————— xyにxとyの積を代入
    PRINT x; "*"; y; "="; xy——— x, y, xyを表示
END SUB
```

【結　果】

```
54.2 * 32.58 = 1765.836
125.33 * 654.1 = 81978.35
```

【例題92】 グローバル変数とローカル変数の働き

xをグローバル，yをローカル変数として，メインプログラムでxとyをまず0とした後10回のくり返しを行い，xとyに各々1を加え表示します．続いてサブルーチンを呼び出し，yを0とした後yに5を加えxとyを表示するプログラムをつくれ．

【解　説】 ◎ グローバル変数とローカル変数の働きについて比べながら確認します．

① メインプログラムでxをグローバルに宣言します．xとyを0とします．yは特別に宣言されていませんからローカルです．

② 10回のくり返しをします．xとyに1を加えます．そして表示します．最初は1，1です．サブルーチンへ分岐し，yを0とし，y+5とした後xとyを表示します．xはグローバルですから1が生きていて1，yはローカルですから5となります．したがって，1回目は1，1，1，5です．

③ 2回目はxは1に1が加えられて1，メインプログラムのyは元の1が生きていて1+1で2です．サブルーチンではxは2，yはまず0とした後5を加えますから5です．したがって，2，2，2，5となります．

【プログラム例】

```
DECLARE SUB ss () ——— サブルーチンの宣言
COMMON SHARED x ——— xをグローバルに宣言
x = 0 ————————┐ xとyを0にする
y = 0 ————————┘
FOR i = 1 TO 10 ——┬— 10回のくり返し
    x = x + 1 ————┼┐ xとyに1を加える
    y = y + 1 ————┼┘
    PRINT x; y; ——┼— xとyの表示
    CALL ss ——————┼— サブルーチンの呼び出し
NEXT i ———————————┘
END

SUB ss ——————————— サブルーチン ss
    y = 0 ————————— yを0とする
    y = y + 5 ————— yに5を加える
    PRINT x; y ———— xとyの表示
END SUB
```

【結　果】

```
1   1   1   5 ────── 1回目
2   2   2   5 ────── 2回目
3   3   3   5 ────── 3回目
4   4   4   5              ⋮
5   5   5   5
6   6   6   5
7   7   7   5
8   8   8   5
9   9   9   5
10   10   10   5
```

【例題 93】　チェーンしたプログラムで変数を共通に利用

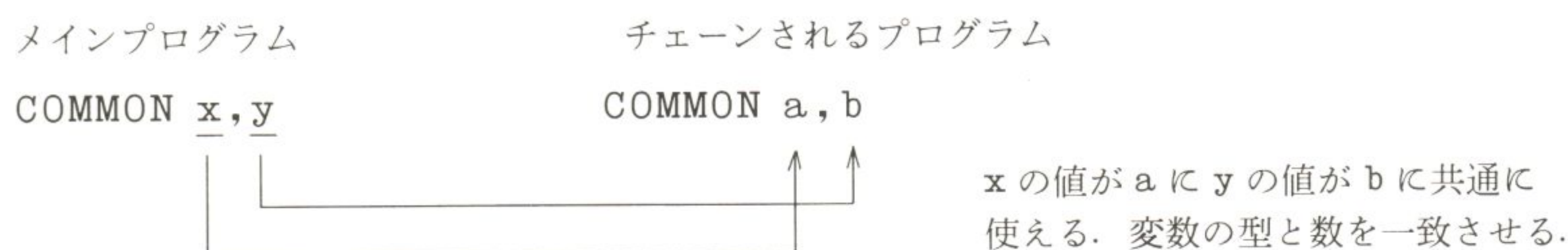
x=123.45，y を 258.32 として x*y を求めて表示した後，ree-95.bas のファイルをチェーンして，x と y の値を生かして商を求めて表示するプログラムをつくれ.

【解　説】　◎　チェーンしたプログラムで変数の値を共通に使います.

① チェーンしたプログラムで変数を共通に使うためにはメインプログラムとチェーンされるプログラムの先頭で COMMON 宣言をします．この時，実数の型と数が大切です．名前は違っていてもかまいません.

メインプログラム　　　　　　　　チェーンされるプログラム

```
COMMON x,y                COMMON a,b
```

x の値が a に y の値が b に共通に使える．変数の型と数を一致させる.

② 本例では x を 123.45，y を 258.32 とした後 x*y を求め，ree-95.bas にチェーンします．x の値は a，y の値は b の値として使えます．したがって，a/b は 123.45/258.32 です.

【プログラム例】

```
COMMON x, y ──────────────── 変数の宣言
x = 123.45 ──────────────┐── x，y の値を代入
y = 258.32 ──────────────┘
seki = x * y ─────────────── 積を求める
PRINT x; "*"; y; "="; seki ── 結果の表示
CHAIN "b:ree-95.bas" ──────── b ドライブの ree-95.bas ファイルへチェーン
END
```

チェーンされるプログラム，ファイル名を"ree-95.bas"として b ドライブに格納しておく

【プログラム例】　(ree-95.bas)

```
COMMON a, b ──────────────── 変数の宣言
shou = a / b ─────────────── 商の計算
PRINT a; "/"; b; "="; shou ── 結果の表示
END
```

【結　果】

```
123.45 * 258.32 = 31889.6
123.45 / 258.32 = .4778956
```

【例題 94】 ローカル変数を一時的に利用

FUNCTION x()を宣言し，次に i=65 から 90 までのくり返しを使って，i のキャラクタコードの文字を表示した後，FUNCTION で i+32 の値を求めそれを FUNCTION の値として，メインプログラムでそれをキャラクタコードとする文字を表示するプログラムをつくれ.

【解　説】 ◎ ローカル変数の値を一時的に利用します.

① i を 65 から 90 までとして，くり返しで CHR$(i)を表示します. 最初は A，最後は Z です.

② FUNCTION を呼び出します. このとき SHARED i によって，ローカル変数である i を一時的に使えるようにします. したがって，x=i+32 とすると FUNCTION の値は最初は 65+32=97 となります. メインプログラムに戻って CHR$(x)を表示すると a となります.

③ 結果は 65，66，…90 をキャラクタコードとする文字 A，B，C…Z の各々の後にその値+32 をキャラクタコードとする文字 a，b，c…z を表示します.

【プログラム例】

```
DECLARE FUNCTION x ()——————FUNCTION の宣言
FOR i = 65 TO 90 ——————————i を 65 から 90 までくり返し
    PRINT CHR$(i); ————————CHR$(i)の表示
    PRINT CHR$(x); ————————FUNCTION の値をキャラクタコードとする文字の表示
NEXT i
END

FUNCTION x ————————————————FUNCTION
    SHARED i ——————————————ローカル変数 i を一時的に使う
    x = i + 32 ————————————i+32 を FUNCTION の値とする
END FUNCTION
```

【結　果】

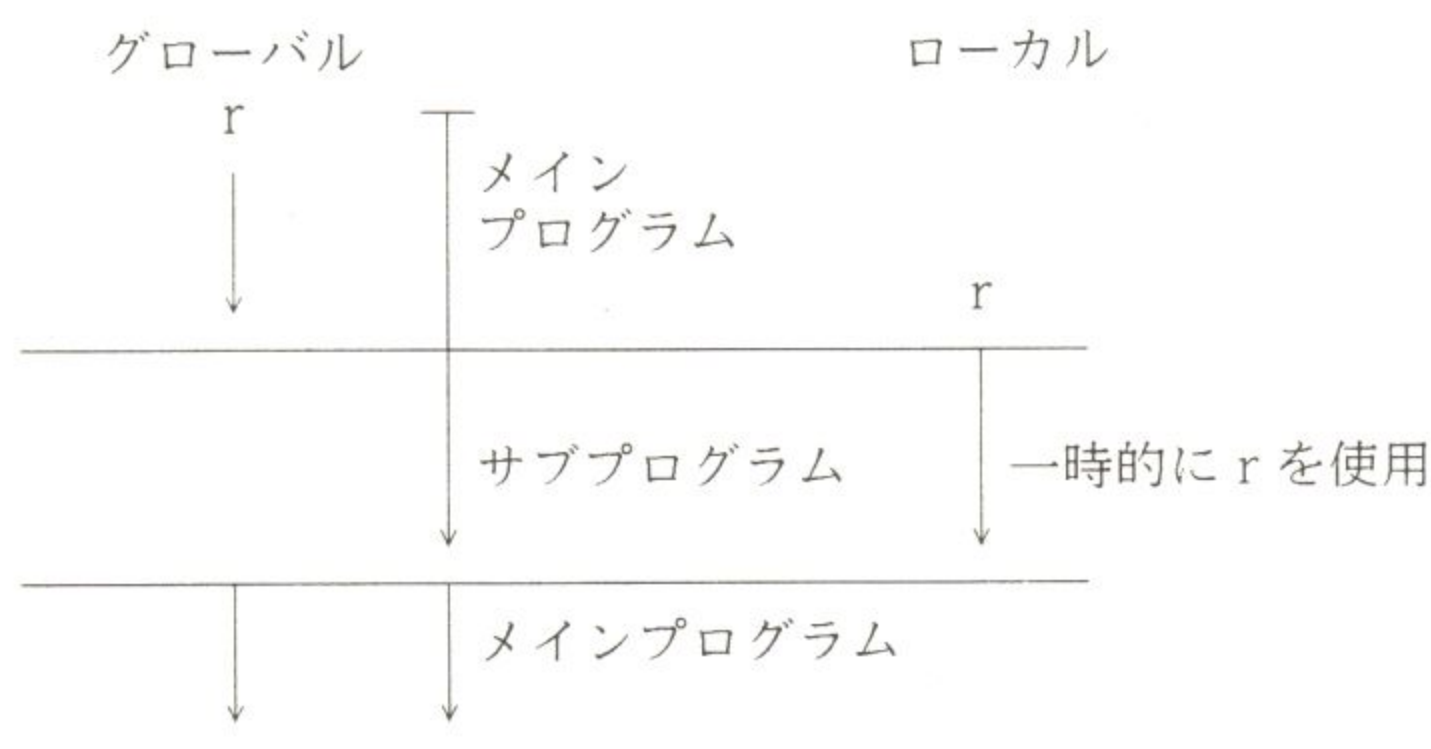
```
AaBbCcDdEeFfGgHhIiJjKkLlMmNnOoPpQqRrSsTtUuVvWwXxYyZz
```

【例題 95】 グローバル変数を一時にローカルで使用

グローバルで定数 pi を 3.14159，変数 r を宣言し，r に値を入力した後サブルーチンを呼び出し，一時的に r をローカル変数として使用し，r を入力して，円周長を求めて表示した後，メインプログラムへもどって元の r を使って円の面積を求めて表示するプログラムをつくれ.

【解　説】 ◎ グローバル変数を一時的にローカルに使います.

① メインプログラムで r に値を入力した後，サブルーチンを呼び出します.

② サブルーチンでは STATIC r と宣言することで r を一時的にサブルーチンの中でのみ使うローカル変数とします. そこで別の r を入力して計算した後メインプログラムへもどります. r の値は元の r の値が残っています.

【プログラム例】

```
DECLARE SUB ss ()———————サブルーチンの宣言
CONST pi = 3.14159——————定数 pi の宣言
COMMON SHARED r ————————r をグローバルに宣言
INPUT r ————————————————r の入力
CALL ss ————————————————サブルーチン ss の呼び出し
s = pi * r * r——————————円の面積を求める
PRINT r; s —————————————r と s の表示
END

SUB ss —————————————————サブルーチン
    STATIC r ———————————グローバル変数 r を一時的にローカルとして使用宣言
    INPUT r ————————————r の入力
    l = 2 * pi * r —————円周長を求める
    PRINT r; l —————————r と l の表示
END SUB
```

【結　果】

```
? 1.5                    1.5 と 2.2 を入力の例
? 2.2
 2.2   13.823
 1.5   7.068578
```

【例題 96】　配列変数

配列 a(1　TO　10)を宣言して下のデータを読み，次にサブルーチンを呼び出し，配列のデータを共用して表示するプログラムをつくれ．

データ；1，5，6，2，7，8，3，6，5，4

【解　説】　◎　配列変数の有効範囲を調べます．

① 配列変数の有効範囲も，普通の変数と同じです．

② 本例では DIM a(1　TO　10)により配列変数 a をローカルに配列宣言します．

③ サブルーチンでは SHARED a( )によって配列変数　a( )を共有します．

【プログラム例】

```
DECLARE SUB ss ()——————————サブルーチンの宣言
DIM a(1 TO 10) ————————————配列変数 a の宣言
```

```
FOR i = 1 TO 10 ─────────┐ 配列へデータを読み込む
    READ a(i)            │
NEXT i ──────────────────┘
CALL ss ───────────────── サブルーチンの呼び出し
END
DATA 1,5,6,2,7,8,3,6,5,4

SUB ss ────────────────── サブルーチン
    SHARED a() ────────── 配列a()を共用
    FOR i = 1 TO 10 ─────┐ a(1)～a(10)を表示
        PRINT a(i);      │
    NEXT i ──────────────┘
END SUB
```

【結　果】

```
1   5   6   2   7   8   3   6   5   4
```

〔演習問題〕

(104)　変数wa，sa，x，yをグローバルに宣言してxとyを入力した後waにx+y，saにx−yを求めてからサブルーチンに分岐して結果を表示するプログラムをつくれ．

(105)　変数xをグローバルに宣言し，数xを入力してサブルーチンを呼び出しx^3を求めて表示するプログラムをつくれ．

(106)　変数shou%，joyo%，x%，y%をグローバルに宣言して，x%，y%を入力して，商をshou%，剰余をjoyo%に求めてからサブルーチンを呼び出し結果を表示するプログラムをつくれ．

(107)　x%を54，y%を25として，積を求めて表示した後，Bドライブのファイル名ree-105.basへチェーンして，x%とy%の値を共通に使ってx%÷y%の剰余を求めて表示するプログラムをつくれ．

(108)　変数x，yをグローバルに宣言し，x，yを入力してからサブルーチンを呼び出し，x，yを一時的にローカルで使用してx，yを入力し，x^2+y^2を求めて表示した後，メインプログラムで，元のx*yを求めて表示するプログラムをつくれ．

(109)　下のデータを配列a$(1　TO　10)に読み，サブルーチンを呼んで配列a$(1　TO　10)を共用して表示するプログラムをつくれ．

データ;Tokyo，Osaka，Nagoya，Fukui，Sapporo，
Yokohama，Kawasaki，Takamatsu，Fukuoka，Sendai

(110)　iを1から10までとしてくり返しを行い，i*i=の後にi^2を表示するプログラムをつくれ．ただし，メインプログラムのローカル変数iをFUNCTION xで一時的に共用してi^2を求めそれをFUNCTIONの値として使うものとする．

(111)　変数aを整数型，bを長整数型でCOMMON宣言し，aを入力してa^3をbに代入して表示するプログラムをつくれ．

(112)　aを単精度実数型，bを単精度実数型，cを倍精度実数型としてCOMMON宣言し，a，bを

入力して商を c に求め表示するプログラムをつくれ．

〔解　答〕

(104)【プログラム例】

```
DECLARE SUB ss () ──────────── サブルーチンの宣言
COMMON SHARED wa, sa, x, y ──── wa, sa, x, y をグローバルに宣言
INPUT x, y ─────────────────── x と y の入力
wa = x + y ───────────────┐── 和と差の計算
sa = x - y ───────────────┘
CALL ss ───────────────────── サブルーチン ss の呼び出し
END

SUB ss ────────────────────── サブルーチン ss
   PRINT x; "+"; y; "="; wa ──┐─ 和と差の表示
   PRINT x; "-"; y; "="; sa ──┘
END SUB
```

【結　果】

```
? 5,3                    5, 3 を入力の例
 5 + 3 = 8
 5 - 3 = 2
```

(105)【プログラム例】

```
DECLARE SUB ss ()──────── サブルーチン ss の宣言
COMMON SHARED x ───────── x をグローバルに宣言
INPUT x ───────────────── x の入力
CALL ss ───────────────── サブルーチン ss を呼び出す
END

SUB ss ────────────────── サブルーチン ss
   s = x * x * x ──────── x³ を s に代入
   PRINT x; s ─────────── x と s を表示
END SUB
```

【結　果】

```
? 12.54                  12.54 を入力の例
 12.54  1971.935
```

(106)【プログラム例】

```
DECLARE SUB ss () ──────────────── サブルーチンの宣言
COMMON SHARED shou%, joyo%, x%, y% ── グローバルに変数を宣言
INPUT x%, y% ───────────────────── x%, y%の入力
shou% = x% ¥ y% ────────────────┐─ 商と剰余を求める
joyo% = x% MOD y% ──────────────┘
CALL ss ────────────────────────── サブルーチンを呼び出す
END
```

```
SUB ss ─────────────────────────────── サブルーチン
    PRINT x%; "/"; y%; "="; shou%; "ｱﾏﾘ "; joyo% ─── 結果の表示
END SUB
```

【結　果】

```
? 125,51                        125 と 51 を入力の例
 125 / 51 = 2 ｱﾏﾘ  23
```

(107)【プログラム例】

```
COMMON x%, y% ──────────────── x%, y%の宣言
x% = 54 ─────────────────────┬ x%, y%に値を代入
y% = 25 ─────────────────────┘
seki% = x% * y% ────────────── 積を seki%に代入
PRINT x%; "*"; y%; "="; seki% ── 結果の表示
CHAIN "b:ree-105.bas"────────── ファイル ree-105 へチェーン
END
```

【プログラム例】（ree-105 のプログラム）

```
COMMON a%, b% ──────────────────── a%, b%の宣言
amari% = a% MOD b% ─────────────── 剰余の計算
PRINT a%; "/"; b%; "ﾉ ｱﾏﾘ="; amari% ─── 結果の表示
END
```

【結　果】

```
54 * 25 = 1350
54 / 25 ﾉ ｱﾏﾘ= 4
```

(108)【プログラム例】

```
DECLARE SUB ss () ──────────────── サブルーチンの宣言
COMMON SHARED x, y ─────────────── x, y をグローバルに宣言
INPUT x, y ─────────────────────── x, y の入力
CALL ss ────────────────────────── サブルーチンを呼び出す
s = x * y ────────────────────────┬ x*y の値の表示
PRINT x; "*"; y; "="; s ──────────┘
END

SUB ss ─────────────────────────── サブルーチン
    STATIC x, y ────────────────── グローバル変数 x, y を一時的にローカルで使用
    INPUT x, y ─────────────────── x, y を入力
    s = x * x + y * y ────────────┬ x²+y² を求めて表示
    PRINT x; "^2+"; y; "^2="; s ──┘
END SUB
```

【結　果】

```
? 12.3,2.5                      12.3, 2.5, 1.35, 8.23 を入力の例
? 1.35,8.23
 1.35 ^2+ 8.23 ^2= 69.5554
 12.3 * 2.5 = 30.75
```

(109)【プログラム例】

```
DECLARE SUB ss () ──────── サブルーチンの宣言
DIM a$(1 TO 10)─────────── 配列の宣言
FOR i = 1 TO 10 ─────────┐ 配列にデータを読み込む
    READ a$(i)           │
NEXT i ──────────────────┘
CALL ss ────────────────── サブルーチンを呼び出して実行
END
DATA Tokyo,Osaka,Nagoya,Fukui,Sapporo
DATA Yokohama,Kawasaki,Takamatsu,Fukuoka,Sendai

SUB ss ─────────────────── サブルーチン
    SHARED a$()─────────── 配列 a$を共用する
    FOR i = 1 TO 10 ─────┐ a$の表示
        PRINT a$(i)      │
    NEXT i ──────────────┘
END SUB
```

【結　果】

```
Tokyo
Osaka
Nagoya
Fukui
Sapporo
Yokohama
Kawasaki
Takamatsu
Fukuoka
Sendai
```

(110)【プログラム例】

```
DECLARE FUNCTION x () ───────── FUNCTION の宣言
FOR i = 1 TO 10 ─────────────── i を 1 から 10 までくり返し
    PRINT i; "*"; i; "=";─────┐ i×i=の後に i² を表示
    PRINT x                   │
NEXT i ───────────────────────┘
END

FUNCTION x ──────────────────── FUNCTION x
    SHARED i ────────────────── ローカル変数 i を一時的に共用で使う
    x = i * i ───────────────── i² を FUNCTION の値とする
END FUNCTION
```

【結　果】

```
1 * 1 = 1
2 * 2 = 4
3 * 3 = 9
4 * 4 = 16
5 * 5 = 25
6 * 6 = 36
7 * 7 = 49
8 * 8 = 64
9 * 9 = 81
10 * 10 = 100
```

(111)【プログラム例】

```
COMMON a AS INTEGER
COMMON b AS LONG
INPUT a
b = a * a * a
PRINT a
PRINT b
END
```

【結　果】

```
? 8
 8
 512
```

(112)【プログラム例】

```
COMMON a AS SINGLE
COMMON b AS SINGLE
COMMON c AS DOUBLE
INPUT a, b
c = a / b
PRINT a; b
PRINT c
END
```

【結　果】

```
? 2354.25,12.85436
 2354.25  12.85436
 183.1479797363281
```

14章　算術関数

SIN, COS, TAN, ATN, SQR, ABS, LOG, EXP, CINT, CLNG, INT, FIX, SGN, RANDOMIZE, RND, TIMER

【例題 97】　3 角関数

角 30 度の sin, cos, tan の値を求めるプログラムをつくれ.

【解　説】　◎　3 角関数の sin, cos, tan を求めます.

① 3 角関数の値は次の形で求まります. ただし, x はラジアン単位です.

正弦	sin(x)	sin(x/180*3.14159) で x 度の場合が求まる
余弦	cos(x)	cos(x/180*3.14159) で x 度の場合が求まる
正接	tan(x)	tan(x/180*3.14159) で x 度の場合が求まる

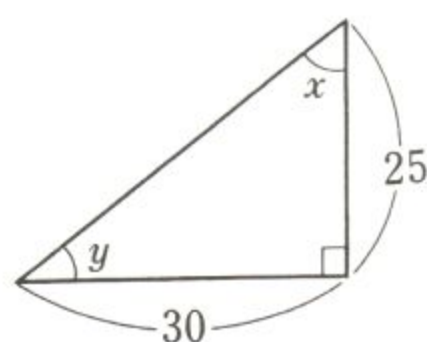
【プログラム例】

```
x = 30 / 180 * 3.14159 ——— 30 度のラジアンの値を x に代入
PRINT SIN(x) ——————————┬— SIN, COS, TAN の値を表示
PRINT COS(x)           │
PRINT TAN(x) ——————————┘
END
```

【結　果】

```
.4999996
.8660256
.5773497
```

【例題 98】　アークタンジェント

下図の角 x 度を求めるプログラムをつくれ.

【解　説】　◎　アークタンジェントを求めます.

① アークタンジェントは次の式で求めます. ただし, 結果はラジアンです.

ATAN(x)

【プログラム例】

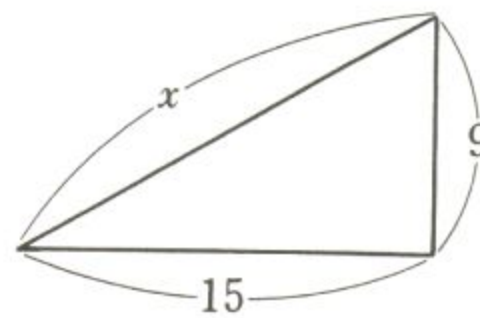
```
xx = ATN(25 / 30)―――――――アークタンジェントの値を求める
x = xx * 180 / 3.14159――――xx ラジアンを x 度に変換する
PRINT x; "ド"
END
```

【結　果】

```
39.8056 ド
```

【例題 99】　平方根

下図の x の長さを求めるプログラムをつくれ.

(図：直角三角形 底辺 15，高さ 9，斜辺 x)

【解　説】　◎　平方根を求めます.

① 平方根は次の形で求めます.

SQR(x)

② したがって，3 角形の斜辺の長さは $\sqrt{15^2+9^2}$ ですから SQR(15^2+9^2)で求められます.

【プログラム例】

```
x = SQR(15 * 15 + 9 * 9)――― √(15²+9²)を求める
PRINT x
END
```

【結　果】

```
17.49286
```

【例題 100】　絶対値

整数を 2 つ入力して，その差を求めるプログラムをつくれ.

【解　説】　◎　絶対値を求めます.

① 2 数の差は，2 数の差の絶対値です．絶対値は次の式で求めます.

ABS(x)

② したがって，a と b の差は次の式で求まります.

ABS(a−b)　　ABS(b−a)でも同じです.

【プログラム例】

```
INPUT a, b ―――――――a と b の入力
PRINT ABS(a - b)――――a と b の差を求める
END
```

【結　果】

```
? 5,2          5と2を入力の例
 3
? 2,5          2と5を入力の例
 3
```

【例題101】　対　数

100万円を年利率4.3%で預金したとき200万円になるのに何年かかるかを求めるプログラムをつくれ.

【解　説】　◎　対数を求めます.

① 100万円を4.3%の年利率で預金したとき y 年で200万円になるとします．すると次の式となります.

$$100*(1+0.043)^{y}=200 \quad \cdots\cdots(1)$$

したがって，
$$(1+0.043)^{y}=2 \quad \cdots\cdots(2)$$

対数をとると，
$$y\log(1.043)=\log 2$$

したがって，
$$y=\log 2/\log(1.043)$$

② 対数は log(x)で表わされます.

したがって，ylog(1.043)=log(2)となり，

$$y=\frac{\log(2)}{\log(1.043)}$$

で y が求まります.

なお，y 年は整数ですから切りあげで y を求めます.

③ なお，常用対数と自然対数は次の関係で求められます.

$$\log_{10}x=\frac{\log_{e}x}{\log_{e}10}$$

LOG(x)は e を底とする自然対数の値です.

④ LOG(x)の x は正の値で，x が単精度実数のときは結果も単精度実数，x が倍精度実数のとき結果も倍精度実数となります.

⑤ e の値はおよそ2.718282です.

【プログラム例】

```
y = LOG(2) / LOG(1.043)———— y年を求める
PRINT INT(y + .99999)————— y年を切りあげる
END
```

【結　果】

```
17
```

【例題 102】　指数関数

$e^{3.87}$ を求めるプログラムをつくれ.

【解　説】　◎　指数関数を求めます.

① $e^{3.87}$ は EXP(3.87)で求められます.

【プログラム例】

```
y = EXP(3.87)———指数関数値を求める
PRINT y
END
```

【結　果】

```
47.94238
```

【例題 103】　四捨五入，切りあげ，切りすて

15.38 と 358.29645 を各々小数点第 2 位と第 4 位で四捨五入，切りあげ，切りすてで各々小数第 1 位と第 3 位まで求めるプログラムをつくれ.

【解　説】　◎　四捨五入，切りあげ，切りすてをします.

① 四捨五入は CINT，CLNG を使います．これらは小数点以下を四捨五入して，それぞれ整数，長整数とします.

② 任意の桁で四捨五入するには次のようにします．CINT を例とします．10^n を掛けたり，割ったりした後 CINT を行い，その後 10^n を掛けたり，割ったりして元へ戻します.

```
15.38      CINT(15.38)→15
           CINT(15.38*10)/10→15.4
           CINT(15.38÷10)*10→20
```

③ 切りすては int を使います．int は切りすてで整数化します．任意の桁位置での切りすては四捨五入と同じ考えです.

```
15.38      INT(15.38)→15
           INT(15.38*10)/10→15.3
           INT(15.38÷10)*10→10
```

④ 切りあげは int を応用します．切りあげを行う桁が 1 以上のときに切りあげますから，その桁に 9 を加えてから切りすてすれば同じ結果が得られます.

```
15.38      INT(15.38+0.9)→16
           INT(15.38*10+0.9)/10→15.4
           INT(15.38÷10+0.9)*10→20
```

【プログラム例】

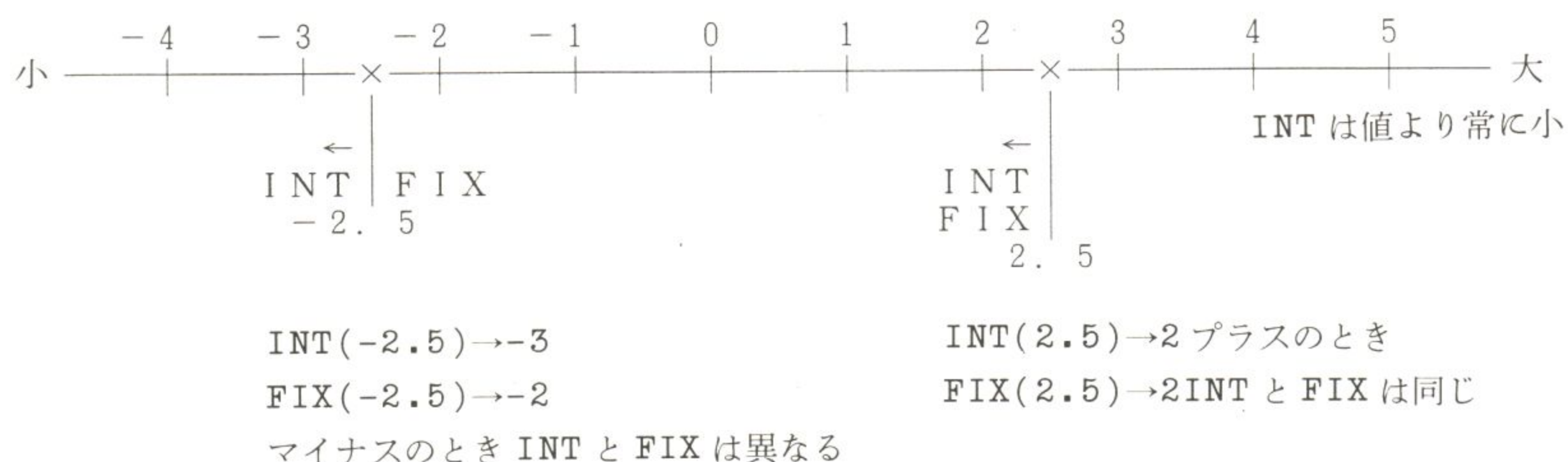
```
x = 15.38
y# = 358.29645#
x1 = CINT(x * 10) / 10 ———————— 小数第 2 位で四捨五入
x2 = INT(x * 10 + .9) / 10 ———— 小数第 2 位で切りあげ
x3 = INT(x * 10) / 10 ————————— 小数第 2 位で切りすて
y1 = CLNG(y# * 1000) / 1000 ——— 小数第 4 位で四捨五入
y2 = INT(y# * 1000 + .9) / 1000 — 小数第 4 位で切りあげ
y3 = INT(y# * 1000) / 1000 ———— 小数第 4 位で切りすて
PRINT x1, x2, x3 ——————————┬— 結果の表示
PRINT y1, y2, y3 ——————————┘
END
```

【結　果】

```
15.4          15.4          15.3
358.296       358.297       358.296
```

【例題 104】　整数部分の値のとり出し

-35.86 の FIX 値と 15.38 の INT 値をとり出すプログラムをつくれ.

```
      −4   −3    −2   −1   0   1   2    3   4   5
小 ——+————+——×——+————+————+———+———+——×——+————+————+—— 大
                                               INT は値より常に小
          ←                              ←
       INT | FIX                        INT
        −2. 5                           FIX
                                         2. 5

INT(-2.5)→-3                          INT(2.5)→2 プラスのとき
FIX(-2.5)→-2                          FIX(2.5)→2INT と FIX は同じ
マイナスのとき INT と FIX は異なる
```

【解　説】　◎　整数部の値をとり出します.

① FIX は INT と整数部をとり出す点で似ていますが, マイナスの場合では異なります. INT はその値をこえない最大の整数値です. FIX は整数部の値です.

【プログラム例】

```
x = -35.86
y = 15.38
x1 = FIX(x)——— -35.86 の整数部の値
y1 = INT(y)——— 15.38 をこえない最大の整数値
PRINT x1, y1
END
```

【結　果】

```
-35          15
```

【例題 105】　SGN

$y=ax^2+3x+c$ の式で a の値を入力し，プラスのとき"シタ　ガ　トツ"，a<0 のとき"ウエ　ガ　トツ"，a=0 のとき"チョクセン"と表示するプログラムをつくれ.

【解　説】　◎　SGN は値の範囲によって 1，0，−1 を与えます.

① 2次式の値は a の値がプラス，マイナス，0 によって次図のようになります.

$a>0$　　$a<0$　　$a=0$

② a を入力し，SGN(a)によって，a がプラスのとき SGN(a)は 1，マイナスのとき−1，0 のとき 0 となります．したがって，a の値によって SGN の値を 3 つに分けて分岐します.

【プログラム例】

```
INPUT a ──────────────── a の入力
s = SGN(a) ───────────── a の SGN を s に求める
IF s > 0 THEN ────────── s の値によって表示を選ぶ
        PRINT "ｼﾀ ｶﾞ ﾄﾂ"
        ELSE
        IF s = 0 THEN
                PRINT "ﾁｮｸｾﾝ"
                ELSE
                PRINT "ｳｴ ｶﾞ ﾄﾂ"
        END IF
END IF
END
```

【結　果】

```
? 5             5 を入力の例
ｼﾀ ｶﾞ ﾄﾂ
? -4            -4 を入力の例
ｳｴ ｶﾞ ﾄﾂ
? 0             0 を入力の例
ﾁｮｸｾﾝ
```

【例題 106】　乱　数

1000〜9999 の乱数を 100 個つくり，1000 台，2000 台，…9000 台が各々いくつできたかを求めるプログラムをつくれ.

【解　説】　◎　乱数をつくります.

① 乱数は RND でつくりますが，これだけでは同じ乱数系列を使うため，プログラムを実行するごとに同じ値がくり返しできます．これは，シミュレーションなどで，システムの構成を変えて再びトライする場合には有効ですが，ゲームでは同じことのくり返しでおもしろくありません．乱数系列を変える命令が RANDOMIZE です.

② RANDOMIZE は実行されると数値を聞いてきて，値を入力するとその値の乱数系列をつくります．つくられる乱数は新しい系列となります．

③ 自動的に新しい乱数系列をつくりたいときは TIMER を利用して，刻々かわる時間の数値を利用してつくり出します．

④ RND(x)は x の値によって 0 から 1 未満の乱数をつくります．x が正または RND のみのときは次の乱数をつくります．0 のときは 1 つ前の乱数，負のときは同じ乱数をつくります．

RND(5)の値を a とする

RND(3)の値を b とする

RND(4)の値を c とする

RND(0)は b となる

RND(5)の値を d とする

RND(-3)は d となる

⑤ 次に乱数は 0 から 1 未満の値ですからこれを使って 1000 から 9999 の数にします．この方法は次のとおりです．

INT(RND(5)*9000)+1000

（INT(RND(5)*9000)：0 から 8999 の値となる）

一般化すると a から b の間の乱数をつくるときは次のようになります．

INT(RND(5)*(b-a+1))+a

【プログラム例】

```
DIM z%(1 TO 9)
RANDOMIZE (TIMER)          ─┐ 100 個の 1000～9999 の乱数を
FOR i = 1 TO 100            │ つくりカウントする
    x% = INT(RND(5) * 9000) + 1000
    y% = INT(x% / 1000)     │
    z%(y%) = z%(y%) + 1     │
NEXT i                     ─┘
FOR i = 1 TO 9             ─┐ 結果の表示
    PRINT i * 1000; "ﾀﾞｲ="; z%(i)
NEXT i                     ─┘
END
```

【結　果】

例①

```
1000 ﾀﾞｲ= 13
2000 ﾀﾞｲ= 12
3000 ﾀﾞｲ= 9
4000 ﾀﾞｲ= 8
5000 ﾀﾞｲ= 13
6000 ﾀﾞｲ= 7
7000 ﾀﾞｲ= 13
8000 ﾀﾞｲ= 10
9000 ﾀﾞｲ= 15
```

例②

```
1000 ﾀﾞｲ= 10
2000 ﾀﾞｲ= 11
3000 ﾀﾞｲ= 9
4000 ﾀﾞｲ= 11
5000 ﾀﾞｲ= 14
```

```
6000 ﾀﾞｲ= 9
7000 ﾀﾞｲ= 11
8000 ﾀﾞｲ= 15
9000 ﾀﾞｲ= 10
```

【例題 107】　乱数を使ったシミュレーション

3 割 2 分 5 厘の成績の打者の今後 30 打席のヒットの確率を求めるプログラムをつくれ.

【解　説】　◎　乱数を使ったシミュレーション

① 乱数を使ったシミュレーションを行う例を 1 つ示します.

② 3 割 2 分 5 厘の打率の打者のヒットの確率は .325 です. したがって, 乱数が 0〜1 未満とすれば乱数をつくって 0〜 .325 のときはヒット, それ以外ではアウトとすればよいことになります.

③ 乱数を 30 回つくって, 0〜 .325 の数を数え, 30 で割れば, 今後 30 打席の打率となります.

【プログラム例】

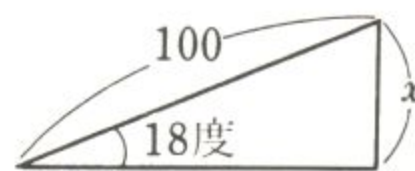
```
RANDOMIZE (TIMER)            ─ 乱数をつくり, 325 以下の
FOR i = 1 TO 30                 とき数を数える
    x = RND(5)
    IF x <= .325 THEN d = d + 1
NEXT i
PRINT d / 30                 ─ 打率の計算
END
```

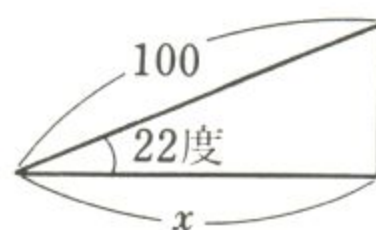
【結　果】

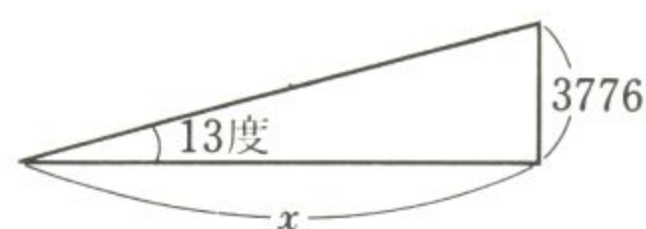
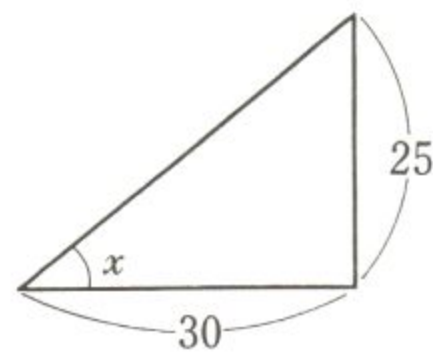
```
.2666667          〔例〕5 回実行したもの
.4333333
.2
.3
.2333333
```

〔演習問題〕

(113)　下図の x を求めるプログラムをつくれ.

100, 18度, x

(114)　下図の x を求めるプログラムをつくれ.

100, 22度, x

(115)　下図の x を求めるプログラムをつくれ.

3776, 13度, x

(116)　下図の x 度と y 度を求めるプログラムをつくれ.

(117)　$3x^2+2x-3=0$ の式での x の値を求めるプログラムをつくれ.

(118)　小数点以下 3 桁の数を 1 つ入力し，小数第 3 位を四捨五入，切りあげ，切りすてで小数点以下第 2 位までの値を求めるプログラムをつくれ.

(119)　150 万円を年利率 5.1%の複利預金をしたとき，元利合計が 3 倍となるのに必要な年数を求めるプログラムをつくれ.

(120)　基準地からの距離を 2 つ入力して，その地区間の距離を求めるプログラムをつくれ.

(121)　サイコロを 10 回ふって 1～6 の目の出かたをシミュレートするプログラムをつくれ.

〔解　答〕

(113)【プログラム例】

```
x = 18 / 180 * 3.14159 ———18 度のラジアン値
PRINT 100 * SIN(x) —————x ラジアンの SIN 値*100
END
```

【結　果】

```
30.90168
```

(114)【プログラム例】

```
x = 22 / 180 * 3.14159 ———22 度のラジアン値
PRINT 100 * COS(x) —————x ラジアンの COS 値*100
END
```

【結　果】

```
92.7184
```

(115)【プログラム例】

```
x = 13 / 180 * 3.14159 ———13 度のラジアン値
PRINT 3776 / TAN(x) —————x ラジアンの TAN 値*3776
END
```

【結　果】

```
16355.67
```

(116)【プログラム例】

```
x = ATN(30 / 25) ─┐アークタンジェント
y = ATN(25 / 30) ─┘
x1 = x * 180 / 3.14159 ─┐x, yラジアンを度に変換
y1 = y * 180 / 3.14159 ─┘
PRINT x1; "ド"
PRINT y1; "ド"
END
```

【結　果】

```
50.19447 ド
39.8056 ド
```

(117)【プログラム例】

```
z = SQR(2 * 2 - 4 * 3 * (-3)) ───── √(b²-4ac)
x1 = (-2 + z) / (2 * 3) ───── (-b+√(b²-4ac))/2a
x2 = (-2 - z) / (2 * 3) ───── (-b-√(b²-4ac))/2a
PRINT x1, x2
END
```

【結　果】

```
.7207592        -1.387426
```

(118)【プログラム例】

```
INPUT x ───────────────── xを入力
x1 = CINT(x * 100) / 100 ───── 小数第3位で四捨五入
x2 = INT(x * 100 + .9) / 100 ─── 小数第3位で切りあげ
x3 = INT(x * 100) / 100 ────── 小数第3位で切りすて
PRINT x1, x2, x3
END
```

【結　果】

```
? 1.354                                  1.354を入力の例
 1.35          1.36          1.35
? 12.459                                 12.459を入力の例
 12.46         12.46         12.45
```

(119)【プログラム例】

```
y = LOG(3) / LOG(1.051) ──── y年を求める
PRINT INT(y + .99999) ───── y年を切りあげる
END
```

【結　果】

```
23
```

(120) **【プログラム例】**

```
INPUT a, b
PRINT "a="; a
PRINT "b="; b                        a, b の入力
PRINT "ｷｮﾘ="; ABS(a - b)             a と b の差を求める
END
```

【結　果】

```
? 152,987            152 と 987 を入力の例
a= 152
b= 987
ｷｮﾘ= 835
? 562,157            562 と 157 を入力の例
a= 562
b= 157
ｷｮﾘ= 405
```

(121) **【プログラム例】**

```
RANDOMIZE (TIMER)
y = 1 / 6
FOR i = 1 TO 10
    x = RND(5)
    IF x <= y THEN PRINT 1;
    IF x > y AND x <= 2 * y THEN PRINT 2;
    IF x > 2 * y AND x <= 3 * y THEN PRINT 3;
    IF x > 3 * y AND x <= 4 * y THEN PRINT 4;
    IF x > 4 * y AND x <= 5 * y THEN PRINT 5;
    IF x > 5 * y THEN PRINT 6;
NEXT i
PRINT
END
```

FOR～NEXT：1～6 を乱数でつくり各々を表示する

【結　果】

```
5  6  1  3  6  3  3  5  5  6        3 回実行した例
5  1  4  5  1  6  2  2  2  4
4  3  5  2  4  4  6  2  2  6
```

15章 論理演算

AND, OR, XOR, EQV, IMP, NOT

【例題 108】 AND

15と3のAND, -15と3のANDを求めるプログラムをつくれ.

【解 説】 ◎ ANDを求めます.

① 論理演算はビット操作をするときにも使います. 16ビットで先頭を符号とし, プラスのとき0, マイナスのとき1とします.

② 15, 3を16ビット, 先頭を符号で表わすと

	符号	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}
A=15	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
B=3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

となります. したがって,

A AND B	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

となります. したがって, 15 AND 3は, 3です.

③ -15, 3を16ビット, 先頭を符号で表わすと

	符号	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}
A=-15	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	1
B=3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

となります. したがって,

A AND B	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

となります. したがって, -15 AND 3は, 1です.

マイナスの値は補数で表わされます.

【プログラム例】

```
x1% = 15
y1% = 3
x2% = -15
PRINT x1% AND y1%―――15と3のAND
PRINT x2% AND y1%―――-15と3のAND
END
```

【結 果】

```
3
1
```

【例題 109】　**OR**

15 と 3 の OR と-15, 3 の OR を求めるプログラムをつくれ.

【解　説】　◎　OR を求めます.

① 15 および 3 を 16 ビットで, 先頭を符号で表わすと

	符号	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
A=15	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
B=3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

となります. したがって,

A OR B	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

となります. したがって, 15　OR　3 は, 15 です.

② -15 と 3 を 16 ビット, 先頭を符号で表わすと

	符号	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
A=-15	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	1
B=3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

となります. したがって,

A OR B	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	1

です. これは-13 です.

【プログラム例】

```
x1% = 15
y1% = 3
x2% = -15
PRINT x1% OR y1% ———15 と 3 の OR
PRINT x2% OR y1% ———-15 と 3 の OR
END
```

【結　果】

```
 15
-13
```

【例題 110】　**XOR**

15 と 3 の XOR, -15 と 3 の XOR を求めるプログラムをつくれ.

【解　説】　◎　XOR を求めます.

① 15, 3 を 16 ビット, 先頭を符号で表わすと

	符号	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
A=15	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
B=3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

となります．したがって，

A XOR B	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0

となります．したがって，15　XOR　3は，12です．

② -15と3を16ビット，先頭を符号で表わすと

	符号	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
A=-15	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	1
B=3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

となります．したがって，

A XOR B	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	0

です．これは-14です．

【プログラム例】

```
x1% = 15
y1% = 3
x2% = -15
PRINT x1% XOR y1%―――15と3のXOR
PRINT x2% XOR y1%―――-15と3のXOR
END
```

【結　果】

```
 12
-14
```

【例題111】　EQV

15と3のEQV，-15と3のEQVを求めるプログラムをつくれ．

【解　説】　◎　EQVを求めます．

① 15と3を16ビット，先頭を符号で表わすと

	符号	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
A=15	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
B=3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

となります．したがって，

A EQV B	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	1

です．これは-13です．

② -15，3を16ビット，先頭を符号で表わすと

	符号	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
A=-15	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	1
B=3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

となります．したがって，

A EQV B	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1

となります．したがって，-15　EQV　3は，13です．

【プログラム例】

```
x1% = 15
y1% = 3
x2% = -15
PRINT x1% EQV y1% ——— 15と3のEQV
PRINT x2% EQV y1% ——— -15と3のEQV
END
```

【結　果】

```
-13
13
```

> **【例題112】　IMP**
>
> 15と3のIMP，-15と3のIMPを求めるプログラムをつくれ.

【解　説】　◎　IMPを求めます.

① 15，3を16ビット，先頭を符号で表わすと

	符号	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}
A=15	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
B=3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

となります．したがって，

A IMP B	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	1

です．これは-13です．

② -15と3を16ビット，先頭を符号で表わすと

	符号	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}
A=-15は，	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	1
B=3は，	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

となります．したがって，

A IMP Bは，	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

となります．すなわち，-15　IMP　3は，15です．

【プログラム例】

```
x1% = 15
y1% = 3
x2% = -15
PRINT x1% IMP y1% ——— 15と3のIMP
PRINT x2% IMP y1% ——— -15と3のIMP
END
```

【結　果】

```
-13
 15
```

【例題 113】　NOT

15 の NOT と-15 の NOT を求めるプログラムをつくれ.

【解　説】 ◎ NOT を求めます.

① 15 を 16 ビットで，先頭を符号で表わすと

	符号	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
A=15	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1

です．したがって，

NOT A	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0

です．これは-16 です.

② -15 を 16 ビットで，先頭を符号で表わすと

	符号	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
A=-15	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	1

です．したがって，

NOT A	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0

です．したがって，14 です.

【プログラム例】

```
x1% = 15
x2% = -15
PRINT NOT x1%———15 の NOT
PRINT NOT x2%———-15 の NOT
END
```

【結　果】

```
-16
 14
```

〔演習問題〕

(122)　189 と 25 の AND を求めるプログラムをつくれ.

(123)　29 と-8 の各々の NOT の OR を求めるプログラムをつくれ.

(124)　80 と 73 の IMP と 80 と 73 の EQV の OR を求めるプログラムをつくれ.

(125)　-15 と－85 の XOR を求めるプログラムをつくれ.

〔解　答〕

(122)【プログラム例】

```
x1% = 189
x2% = 25
PRINT x1% AND x2%
END
```

【結　果】

```
25
```

(123)【プログラム例】

```
x1% = 29
x2% = -8
PRINT (NOT (x1%)) OR (NOT (x2%))———(29 の NOT) と (-8 の NOT) の OR
END
```

【結　果】

```
-25
```

(124)【プログラム例】

```
x1% = 80
x2% = 73
PRINT (x1% IMP x2%) OR (x1% EQV x2%)———(80 IMP 73) と (80 EQV 73) の OR
END
```

【結　果】

```
-17
```

(125)【プログラム例】

```
x1% = -15
x2% = -85
PRINT x1% XOR x2%———-15 と-85 の XOR
END
```

【結　果】

```
90
```

16章　文字列関数

LEFT$, RIGHT$, MID$, LEN, SPACE$, LTRIM$, RTRIM$, MID$=, INSTR, LCASE$, UCASE$, STRING$

【例題 114】　文字列のとり出し

"Microsoft Quick Basic"の中から"Microsoft", "Basic", "Quick"をとり出して表示するプログラムをつくれ.

【解　説】　◎　文字列の中から任意の文字列をとり出します.

① a$を"Microsoft Quick Basic"とします.

② LEFT$, RIGHT$, MID$は各々a$の中から文字列をとり出します.

Microsoft	Quick	Basic
LEFT$(a$,9)	MID$(a$,11,5)	RIGHT$(a$,5)
左側 9 文字	11 番目から 5 文字	右側 5 文字

【プログラム例】

```
a$ = "Microsoft Quick Basic"—— 文字列を a$に代入
b$ = LEFT$(a$, 9) ————————— 左側 9 文字を b$に代入
c$ = RIGHT$(a$, 5) ———————— 右側 5 文字を c$に代入
d$ = MID$(a$, 11, 5)——————— 11 番目から 5 文字を d$に代入
PRINT b$ ——————————————┬—— b$, c$, d$の表示
PRINT c$               │
PRINT d$               │
END————————————————————┘
```

【結　果】

```
Microsoft
Basic
Quick
```

【例題 115】　文字列の長さ

"Microsoft Quick Basic"を 1 文字ずつとり出して表示するプログラムをつくれ.

【解　説】　◎　文字列の長さを求めます.

① a$に"Microsoft Quick Basic"を代入します.

② LEN(a$)は文字列 a$の長さ, 文字数を求めます.

【プログラム例】

```
a$ = "Microsoft Quick Basic"———文字列を a$に代入
b = LEN(a$) ——————————————————文字数を b に代入
FOR i = 1 TO b ———————————————1 番目から b 番目まで 1 文字ずつとり
    c$ = MID$(a$, i, 1)        出して c$へ代入し，c$を順に表示
    PRINT c$;
NEXT i ———————————————————┐改行
PRINT ————————————————————┘
END
```

【結　果】

```
Microsoft Quick Basic
```

【例題 116】　空　白

5 桁 3 行目から 5 つの文字列を入力して，6，8，10，12，14 行の 5 桁目からそれぞれ表示するプログラムをつくれ．

【解　説】　◎　空白を表示します．

① 文字列を入力する前に，同じ行列から空白を表示して，入力する位置を空白にします．前回入力した文字列が消え，同じ行列から文字列の入力ができます．

② 空白は space$(x) で x 個の空白を持つ文字列をつくります．

【プログラム例】

```
DIM a$(5) ————————————————————————————5 文字列の入る配列変数の宣言
FOR i = 1 TO 5 ——————————————————┐————5 文字列の入力
    LOCATE 3, 5: PRINT SPACE$(30) │    5 桁 3 行目から 30 空白文字をかく
    LOCATE 3, 5: INPUT a$(i)      │    5 桁 3 行目から文字列の入力
NEXT i ——————————————————————————┘
FOR i = 1 TO 5 ——————————————————┐————表示
    LOCATE 4 + i * 2, 5: PRINT a$(i)│
NEXT i ——————————————————————————┘
END
```

【結　果】

```
? C            左の 5 文字列を入力した例


Microsoft

Quick

Basic

Quick

C
```

【例題 117】　空白の除去

a$を"□Microsoft□", b$を"□Quick□", c$を"□Basic□"（□は空白）として a$+b$+c$と各文字列の左側の空白をとって a$+b$+c$を表示するプログラムをつくれ.

【解　説】　◎　空白を除去します.

① LTRIM$(a$)で文字列 a$の左側（先頭）の空白をとります.

a$	LTRIM$(a$)
"□Microsoft□"	"Microsoft□"

② RTRIM$(a$)は文字列の右側の空白をとります. これは〔演習問題〕(128) で扱います.

【プログラム例】

```
a$ = " Microsoft "─────────┐a$, b$, c$に各文字列を代入
b$ = " Quick "             │
c$ = " Basic "─────────────┘
PRINT a$ + b$ + c$ ──────────a$+b$+c$の表示
PRINT LTRIM$(a$) + LTRIM$(b$) + LTRIM$(c$)───左側の空白を除去して a$+b$+c$を表示
END
```

【結　果】

```
 Microsoft  Quick  Basic
Microsoft Quick Basic
```

【例題 118】　文字列の入れかえ

a$を"Microsoft Quick Basic"とし, 次に"Basic"を"C□□□□"（□は空白）で入れかえて表示するプログラムをつくれ.

【解　説】　◎　文字列の 1 部の入れかえをします.

① a$を"Microsoft Quick Basic"とします. b$を"C□□□□"（□は空白）とします.

② "Basic"の部分を"C□□□□"と入れかえます.

```
Microsoft Quick Basic
                │
              □□□□□
```

MID$(a$,17,5)=b$

a$の 17 番目から 5 文字を b$で入れかえ

【プログラム例】

```
a$ = "Microsoft Quick Basic"───a$に文字列を代入
b$ = "C    "───────────────────b$に文字列を代入
MID$(a$, 17, 5) = b$ ──────────a$の 17 番目から 5 文字を b$で入れかえ
PRINT a$ ──────────────────────a$の表示
END
```

【結　果】

```
Microsoft Quick C
```

【例題 119】　検　索

a$を"Microsoft Quick Basic"，b$を"Microsoft"，c$を"soft"，d$を"Quick"として，a$の最初から b$，c$，d$があるかどうか，また a$の 17 番目以降に d$があるかどうかを検索するプログラムをつくれ．

【解　説】　◎　文字列の検索をします．

① a$を"Microsoft Quick Basic"，b$を"Microsoft"，c$を"soft"，d$を"Quick"とします．

② INSTR(1,a$,b$)は a$の中に b$があるかどうかを検索します．このとき 1 番目からさがします．見つかった場合，それが何番目からあるか，その番号を与えます．"Microsoft Quick Basic"の中に"Microsoft"は 1 番目からありますので INSTR(1,a$,b$)は 1 です．見つからないときは 0 です．

【プログラム例】

```
a$ = "Microsoft Quick Basic"   ─┐文字列を a$～d$に代入
b$ = "Microsoft"                │
c$ = "soft"                     │
d$ = "Quick"                   ─┘
PRINT INSTR(1, a$, b$)         ─┐a$の中で b$，c$，d$を最初から検索，
PRINT INSTR(1, a$, c$)          │
PRINT INSTR(1, a$, d$)         ─┘
PRINT INSTR(17, a$, d$)        ── a$の中で d$を 17 番目から検索
END
```

【結　果】

```
1
6
11
0
```

【例題 120】　小文字化

"Microsoft Quick Basic"を小文字で表示するプログラムをつくれ．

【解　説】　◎　大文字を小文字化します．

① LCASE$(a$)は a$の中の大文字を小文字にします．

② UCASE$(a$)は a$の中の小文字を大文字にします．これは〔演習問題〕(131) で扱います．

【プログラム例】

```
a$ = "Microsoft Quick Basic"   ── 文字列を a$に代入
PRINT LCASE$(a$)               ── a$の大文字を小文字にして表示
END
```

【結　果】

```
microsoft quick basic
```

【例題 121】　特定の文字の列

"A"を 10 個つないだ文字列をつくるプログラムをつくれ.

【解　説】　◎　特定の文字を指定した数つないだ文字列をつくります.

① STRING$(a,b)ではアスキーコードを b とする文字・記号を a 個つないだ文字列をつくります.

STRING$(10,65)

　　　　└── 65 すなわち"A"を 10 個つないだ文字列

STRING$(10,"A")でもよい.

STRING$(10,& H41)でもよい.

② b が文字列の場合，先頭 1 文字となります.

【プログラム例】

```
a$ = STRING$(10, 65) ――― "A"を 10 個つないだ文字列を a$に代入
PRINT a$ ―――――――――――― a$の表示
END
```

【結　果】

```
AAAAAAAAAA
```

【例題 122】　グラフ

次の表のデータをグラフとするプログラムをつくれ.

成績

名前	点
川上	80
山下	90
足立	56
瀬島	74

【解　説】　◎　STRING$を使ってグラフをつくります.

① STRING$を使って簡単なグラフをつくります.

② 100 点を"*"50 個で表わすとすれば各人の点数/2 個の＊をつないだ文字列をつくって表示すれば各人の点を横棒グラフで表わせます.

例

100 点　　STRING$(100/2,"*")　　＊が 50 個の文字列

90 点　STRING$(　90/2,"＊")　＊が 45 個の文字列

x 点　STRING$(　x/2,"＊")　＊が x/2 個の文字列

③ 目盛りと軸は"+"と"-"を 4 つつないだ文字列を 10 個と最後に"+"をつけます.

b$="+"+STRING$(4,"-")とすると b$が"+----"となります.

```
+-----+-----+-----………+-----+
  0 b$   10 b$   20 b$      90 b$     100点を示す
```

【プログラム例】

```
DIM a$(4), s$(4)                      ── a$に名前, s$に点数にあう文字列が入る
FOR i = 1 TO 4                        ── 名前と点数を読み, 名前を a$, 点数にあう
    READ a$(i), x                        "＊"の文字列を s$に代入
    s$(i) = STRING$(x / 2, "*")
NEXT i
b$ = "+" + STRING$(4, "-")            ── b$に"+----"をつくって代入
LOCATE 3, 20: PRINT "4ﾆﾝ ﾉ ｾｲｾｷ(ﾃﾝ)"  ── 標題
FOR i = 1 TO 4                        ── 名前, +, 点数にあう＊を表示
    LOCATE 3 + 2 * i, 10: PRINT a$(i)
    LOCATE 3 + 2 * i, 16: PRINT "+"
    LOCATE 3 + 2 * i, 17: PRINT s$(i)
NEXT i
FOR i = 1 TO 10                       ── 軸の表示
    LOCATE 13, 16 + (i - 1) * 5: PRINT b$
NEXT i
LOCATE 13, 16 + (i - 1) * 5: PRINT "+"
FOR i = 0 TO 100 STEP 10              ── 点数を表示
    LOCATE 14, 15 + i / 10 * 5: PRINT i
NEXT i
END
DATA ｶﾜｶﾐ,80,ﾔﾏｼﾀ,90,ｱﾀﾞﾁ,56,ｾｼﾞﾏ,74
```

【結　果】

```
              4ﾆﾝ ﾉ ｾｲｾｷ(ﾃﾝ)

ｶﾜｶﾐ     +****************************************
ﾔﾏｼﾀ     +*********************************************
ｱﾀﾞﾁ     +****************************
ｾｼﾞﾏ     +*************************************

         +----+----+----+----+----+----+----+----+----+----+
         0    10   20   30   40   50   60   70   80   90   100
```

〔演習問題〕

(126)　"マイクロソフト　クィック　ベーシック"の中から, "マイクロソフト", "ベーシック", "クィック"を各々とり出して表示するプログラムをつくれ.

(127)　文字列を 2 つ a$, b$に代入し, 長い方を表示するプログラムをつくれ.

(128)　a$を"□Microsoft□", b$を"□Quick□", c$を"□Basic□"としてa$+b$+c$と，各々の文字列の右側の空白をとってa$+b$+c$を表示するプログラムをつくれ.

(129)　a$を"ダイレクト□モード□"とし，b$を"ジッコウ"としa$の8番目から5文字をb$で入れかえるプログラムをつくれ.

(130)　a$を"Microsoft Quick Basic"とし，文字列をb$に入力してa$の中にb$があればa$を，なければb$を表示するプログラムをつくれ.

(131)　"Microsoft Quick Basic"を大文字にして表示するプログラムをつくれ.

(132)　#を10個つないだ文字列を表示するプログラムをつくれ.

(133)　"Microsoft"の先頭1文字Mを20個つないだ文字列を表示するプログラムをつくれ.

(134)　下のデータを棒グラフにするプログラムをつくれ.

5社の初任給

社　名	初任給（万円）
中西ＳＳ	15
日本鉄工	19
東京ＳＳ	22
島 商 事	12
本多印刷	10

(135)　"Microsoft"の各文字の後に2スペースをつけて表示するプログラムをつくれ.

(136)　"USA"を画面左から右へ動かすプログラムをつくれ.

〔解　答〕

(126)【プログラム例】

```
a$ = "ﾏｲｸﾛｿﾌﾄ ｸｲｯｸ ﾍﾞｰｼｯｸ"———文字列をa$に代入
b$ = LEFT$(a$, 7) ———————左側7文字をb$に代入
c$ = RIGHT$(a$, 6) ——————右側6文字をc$に代入
d$ = MID$(a$, 9, 4)——————9番目から4文字をd$に代入
PRINT b$ ————————————┬b$, c$, d$の表示
PRINT c$             │
PRINT d$             │
END——————————————————┘
```

【結　果】

```
ﾏｲｸﾛｿﾌﾄ
ﾍﾞｰｼｯｸ
ｸｲｯｸ
```

(127)【プログラム例】

```
INPUT a$ ——————————文字列を入力してa$へ代入
INPUT b$ ——————————文字列を入力してb$へ代入
```

```
l1 = LEN(a$)―――――――a$の文字数を l1 に代入
l2 = LEN(b$)―――――――b$の文字数を l2 に代入
IF l1 >= l2 THEN――┬―長い方の文字列を表示
    PRINT a$      │
ELSE              │
    PRINT b$      │
END IF ――――――――――┘
END
```

【結　果】

```
? length                length と width を入力の例
? width
length
? dog                   dog と window を入力の例
? window
window
```

(128)【プログラム例】

```
a$ = " Microsoft " ―――――――――――┬―a$, b$, c$に各文字列を代入
b$ = " Quick "                 │
c$ = " Basic "―――――――――――――――┘
PRINT a$ + b$ + c$ ――――――――――――a$+b$+c$を表示
PRINT RTRIM$(a$) + RTRIM$(b$) + RTRIM$(c$)――――右側の空白を除去して a$+b$+c$を表示
END
```

【結　果】

```
Microsoft  Quick  Basic
Microsoft Quick Basic
```

(129)【プログラム例】

```
a$ = "ﾀﾞｲﾚｸﾄ ﾓｰﾄﾞ "――――a$に文字列を代入
b$ = "ｼﾞｯｺｳ"―――――――――b$に文字列を代入
MID$(a$, 8, 5) = b$――――a$の 8 番目から 5 文字を b$で入れかえ
PRINT a$ ――――――――――――a$の表示
END
```

【結　果】

```
ﾀﾞｲﾚｸﾄ ｼﾞｯｺｳ
```

(130)【プログラム例】

```
a$ = "Microsoft Quick Basic"――――文字列を a$に代入
INPUT b$ ――――――――――――――――――――文字列を b$に入力
IF INSTR(1, a$, b$) <> 0 THEN ――┬―a$の中に b$があれば a$を表示, なければ b$を表示
    PRINT a$                     │
ELSE                             │
    PRINT b$                     │
END IF ――――――――――――――――――――――――┘
END
```

【結　果】

```
? Basic                          Basicを入力の例
Microsoft Quick Basic
? basic                          basicを入力の例
basic
```

(131)【プログラム例】

```
a$ = "Microsoft Quick Basic"――――文字列をa$に代入
PRINT UCASE$(a$) ――――――――――――a$を大文字にして表示
END
```

【結　果】

```
MICROSOFT QUICK BASIC
```

(132)【プログラム例】

```
a$ = STRING$(10, "#") ――――"#"を10個つないだ文字列をa$に代入
PRINT a$ ――――――――――――――――a$の表示
END
```

【結　果】

```
##########
```

(133)【プログラム例】

```
a$ = STRING$(20, "Microsoft") ――――"M"を20個つないだ文字列をa$に代入
PRINT a$ ――――――――――――――――――――a$の表示
END
```

【結　果】

```
MMMMMMMMMMMMMMMMMMMM
```

(134)【プログラム例】

```
DIM a$(5), s$(5) ――――――――――社名がa$, 初任給にあう*の文字列がs$に入る
FOR i = 1 TO 5 ――――――――――┐データを読んで社名をa$, 初任給にあう*の文字列をs$に代入
    READ a$(i), x              │
    s$(i) = STRING$(x, "*")    │
NEXT i ――――――――――――――――――┘
b$ = "+" + STRING$(9, "-") ――――――――――b$に"+-----"を代入
LOCATE 3, 20: PRINT "5ｼｬ ﾉ ｼｮﾆﾝｷｭｳ(ﾏﾝｴﾝ)"――――標題
FOR i = 1 TO 5 ――――――――――――――――――――┐社名と+と初任給にあう*を表示
    LOCATE 3 + 2 * i, 5: PRINT a$(i)     │
    LOCATE 3 + 2 * i, 16: PRINT "+"      │
    LOCATE 3 + 2 * i, 17: PRINT s$(i)    │
NEXT i ――――――――――――――――――――――――――┘
```

```
FOR i = 1 TO 3 ──────────────────┐─軸の表示
    LOCATE 15, 16 + (i - 1) * 10: PRINT b$
NEXT i                                    │
LOCATE 15, 16 + (i - 1) * 10: PRINT "+"───┘
FOR i = 0 TO 30 STEP 10 ─────────┐─初任給額を表示
    LOCATE 16, 15 + i: PRINT i   │
NEXT i ──────────────────────────┘
END
DATA ﾅｶﾆｼSS,15,ﾆﾎﾝﾃｯｺｳ,19,ﾄｳｷｮｳSS,22
DATA ｼﾏｼｮｳｼﾞ,12,ﾎﾝﾀﾞ ｲﾝｻﾂ,10
```

【結　果】

```
                5ｼｬ ﾉ ｼｮﾆﾝｷｭｳ(ﾏﾝｴﾝ)

ﾅｶﾆｼSS          +***************

ﾆﾎﾝﾃｯｺｳ         +*******************

ﾄｳｷｮｳSS         +**********************

ｼﾏｼｮｳｼﾞ         +************

ﾎﾝﾀﾞ ｲﾝｻﾂ       +**********

                +---------+---------+---------+
                0         10        20        30
```

(135)【プログラム例】

```
a$ = "Microsoft"──────────────────文字列を a$に代入
b = LEN(a$) ──────────────────────b に文字数を代入
FOR i = 1 TO b ───────────────┐──1 文字ずつ後に 2 つの空白をつけて表示
    PRINT MID$(a$, i, 1) + SPACE$(2);
NEXT i ───────────────────────┘
PRINT
END
```

【結　果】

```
M  i  c  r  o  s  o  f  t
```

(136)【プログラム例】

```
a$ = "USA"────────────────────文字列を a$に代入
b$ = SPACE$(3)────────────────3 空白の文字列を b$に代入
FOR i = 1 TO 77 ──────────┐──5 行 1 列から 77 列まで 1 列ごとに USA を表示し，それを消去する
    LOCATE 5, i: PRINT a$ │
    FOR j = 1 TO 500: NEXT j
    LOCATE 5, i: PRINT b$ │
NEXT i ───────────────────┘
END
```

【結　果】

USA　　　　(USA が左から右へ動きます)

17章 漢　字

LEFT$, RIGHT$, KMID$, KMID$=, KLEN, KINSTR, KEXT$, JIS$, CHR$, KTN$

【例題 123】 漢字の表示
”クィックベーシック　を　学びましょう。”を表示するプログラムをつくれ.

【解　説】 ◎ 漢字を表示します.

① 次の形で漢字を表示します.

```
PRINT "クィックベーシック　を　学びましょう。"
```

これは次のようにしても同じです.

```
a$="クィックベーシック　を　学びましょう。"
PRINT a$
```

② 漢字は漢字フロントエンドプロセッサを使います. 本例は”ATOK6”を使用しています. フロントエンドプロセッサの組込みは準備編のとおりです. また, 漢字変換については”ATOK6”を使用しているワープロソフトの説明書によってください. ここでは次の点についてのみふれておきます.

③ CTRL と XFER キーをおします. 漢字変換モードとなります. f･10 キーをおすことで次の5つのモードが選べます.

連ローマ字漢字	ローマ字で入力して漢字変換します
連カナ漢字	カナで入力して漢字変換します
半角英数カナ	半角の英, 数, カナがそのまま入ります
コードJIS	漢字JISコードで漢字を入力します
記号	記号で入力します

④ いずれも終了は CTRL + XFER です.

⑤ 本例についてのキー操作を示します.

PRINT”の後 CNTR + XFER をおします. f･10 キーを4回おして, 記号モードとします. ↑ ↓ キーをおして, ”ク”を含む行を表示させます. ”ク”が反転しています. ↲ します. 同様の手順で矢印キーと ↲ キーで”イックベーシック”まで入力します. [　　] キーをおして空白をつくり ↲ します. f･10 キーで連ローマ字漢字モードとし, W O とします. [　　] キーをおし ↲ します. ”を□”が表示されます. M A N A B I M A S H O U とキーをおした後, XFER をおします. ”学びましょう”となります. ↲ で確定します. f･10 キーを3回おしてコードJISモードとして, 2123 ↲ します (2123は”。”のJISコードです).

最後に CTRL + XFER で漢字変換モードは終了です.

”を記入して, PRINT 命令ができました.

【プログラム例】

```
PRINT "クィックベーシック を 学びましょう。"
END
```

【結　果】

クィックベーシック を 学びましょう。

【例題 124】 漢字文字列の左側の文字のとり出し

”クイックベーシック を 学びましょう。”の中から”クイック”をとり出して表示するプログラムをつくれ.

【解　説】 ◎ 漢字文字列の中から左側の文字列をとり出します.

① a$を”クイックベーシック を 学びましょう。”とします.

② LEFT$はa$の中から左側の指定した文字列をとり出します.

③ 漢字の場合は1文字が2バイトです. したがって, 次のように8の指定で4文字をとり出します.

クイックベーシック を 学びましょう。

LEFT$(a$,8)

左側4文字

【プログラム例】

```
a$ = "クィックベーシック を 学びましょう。"
b$ = LEFT$(a$, 8)
PRINT b$
END
```

【結　果】

クィック

【例題 125】 漢字文字列の右側の文字列

”クイックベーシック を 学びましょう。”の中から右側7文字をとり出して表示するプログラムをつくれ.

【解　説】 ◎ 漢字文字列の右側7文字をとり出します.

① a$を”クイックベーシック を 学びましょう。”とします.

② 次の形で右側7文字をとり出します.

RIGHT$ (a$,14)

RIGHT$: 右側の文字列

a$: 元の文字列

14: 7文字. 漢字では1文字を2と数える

【プログラム例】

```
a$ = "クィックベーシック を 学びましょう。"
b$ = RIGHT$(a$, 14)
PRINT b$
END
```

【結　果】

学びましょう。

【例題126】　漢字文字列の中間の文字列

"クイックベーシック　を　学びましょう。"の中から5文字目から5文字をとり出して表示するプログラムをつくれ.

【解　説】　◎　漢字文字列の中間の文字列をとり出します.

① a$と"クイックベーシック　を　学びましょう。"とします.

② 次の形で中間の文字列をとり出します.

```
KMID$    (a$,5,5)
          ‾‾ ‾ ‾
中間の文字列  |  |  └── 5文字. KMID$では漢字1文字を1と数える
          |  └── 5文字目から
          元の文字列
```

この例では"ベーシック"となります.

【プログラム例】

```
a$ = "クィックベーシック を 学びましょう。"
b$ = KMID$(a$, 5, 5)
PRINT b$
END
```

【結　果】

ベーシック

【例題127】　文字列の長さ

"クイックベーシック　を　学びましょう。"を1文字ずつとり出して表示するプログラムをつくれ.

【解　説】　◎　文字列の長さを求めます.

① a$に"クイックベーシック　を　学びましょう。"を代入します.

③ KLEN(a$)は文字列a$の長さ，文字数を求めます. KLENは漢字を1文字を1と数えます.

【プログラム例】

```
a$ = "クィックベーシック を 学びましょう。"
b = KLEN(a$)
FOR i = 1 TO b
```

```
    PRINT KMID$(a$, i, 1);
NEXT i
PRINT
END
```

【結　果】

クィックベーシック　を　学びましょう。

【例題 128】　文字列の入れかえ

a$を"平成元年12月31日"とし，b$に"11月15日"を代入し，"12月31日を"11月15日"で入れかえて表示するプログラムをつくれ.

【解　説】　◎　漢字文字列の1部の入れかえをします.

① a$を"平成元年12月31日"とします．b$を"11月15日"とします.

② "12月31日"の部分を"11月15日"と入れかえます.

平成元年12月31日
↑
11月15日

KMID$(a$,5,6)=b$
a$の5番目から6文字をb$で入れかえ

【プログラム例】

```
a$ = "平成元年１２月３１日"
b$ = "１１月１５日"
KMID$(a$, 5, 6) = b$
PRINT a$
END
```

【結　果】

平成元年１１月１５日

【例題 129】　検　索

a$を"クイックベーシック　を　学びましょう。"，b$を"学びましょう"，c$を"クイックC"として，a$の最初からb$，c$があるかどうかを検索するプログラムをつくれ.

【解　説】　◎　漢字文字列の検索をします.

① a$を"クイックベーシック　を　学びましょう。"，b$を"学びましょう"，c$を"クイック　C"とします.

② KINSTR(1,a$,b$)はa$の中にb$があるかどうかを検索します．このとき1番目からさがします．KINSTRは漢字1文字を1と数えて検索します．見つかった場合，それが何番目からあるか，その番号を与えます．"クイックベーシック　を　学びましょう。"の中に"学びましょう"は13番

目からありますので INSTR(1,a$,b$)は13です．見つからないときは0です．

【プログラム例】

```
a$ = "クィックベーシック を 学びましょう。"
b$ = "学びましょう"
c$ = "クィック C"
PRINT KINSTR(1, a$, b$)
PRINT KINSTR(1, a$, c$)
END
```

【結　果】

```
13
0
```

【例題 130】　全角文字と半角文字

"A はアルファベットの最初の文字です"の半角文字と全角文字をわけて表示するプログラムをつくれ．ただし，"A"は半角文字，他は全角文字とします．

【解　説】　◎　半角文字と全角文字をとり出します．

① KEXT$(a$,0)はa$の中の半角文字をとり出します．この例では"A"のみです．

② KEXT$(a$,1)はa$の中の全角文字をとり出します．

【プログラム例】

```
a$ = "Aはアルファベットの最初の文字です"
b$ = KEXT$(a$, 0)
c$ = KEXT$(a$, 1)
PRINT b$
PRINT c$
END
```

【結　果】

```
A
はアルファベットの最初の文字です
```

【例題 131】　漢字の JIS コード

"世界"の"世"の JIS コードを表示するプログラムをつくれ．

【解　説】　◎　漢字の JIS コードを求めます．

① JIS$(a$)では文字列 a$の先頭文字が2バイト文字のときその文字の16進のJISコードを4バイトのアスキー文字列で求めます．1バイト文字のときはその文字の10進のJISコードを3バイト以下のアスキー文字列で求めます．

② a$を"世界"とすると"世"のJISコードで4024です．

③ a$を"A"とすると65，a$を"a"とすると97です．

【プログラム例】

```
a$ = "世界"
PRINT JIS$(a$)
END
```

【結　果】

```
4024
```

【例題 132】　漢字の CHR$

CHR$を使って"世界"と表示するプログラムをつくれ.

【解　説】　◎　シフト JIS コードで漢字を表示します.

① CHR$(3173)で"世", CHR$(1953)で"界"を持ちます.

② 0 から 255 のときは 1 バイト文字, 256 から 11,535 のときはシフト JIS に対応する 2 バイト文字を与えます.

【プログラム例】

```
PRINT CHR$(3173);
PRINT CHR$(1953)
END
```

【結　果】

```
世界
```

【例題 133】　漢字文字列の句点コード

"ABCD 大文字 abcd 小文字"の各文字の句点コードを表示するプログラムをつくれ.

【解　説】　◎　句点コードを求めます.

① KTN$(c$)は文字 c$が 2 バイト文字のときその文字に対応する句点コードを持ちます. 1 バイト文字のときはアスキーコードを持ちます. "A"は 65, "B"は 66, "大"は 3471 などです.

【プログラム例】

```
a$ = "ABCD大文字abcd小文字"
b = KLEN(a$)
FOR i = 1 TO b
    c$ = KMID$(a$, i, 1)
    PRINT KTN$(c$); "  ";
NEXT i
PRINT
END
```

【結　果】

```
65  66  67  68  3471  4224  2790  97  98  99  100  3014  4224  2790
```

【例題 134】　漢字の入力

漢字文字列を入力して，表示するプログラムをつくれ．

【解　説】　◎　漢字の入力をします．

① 漢字の入力も，普通の文字列の入力と同様に INPUT x$で行います．

② プログラムが実行されると?を表示し，入力を要求しますので，CTRL キーと XFER キーをおして漢字入力モードとして，漢字を入力します．

③ 入力が決まったら CTRL キーと XFER キーを再度おして，漢字入力モードを解除します．

【プログラム例】

```
INPUT x$
PRINT x$
END
```

【結　果】

```
? おおさか
おおさか
? 東京
東京
```

〔演習問題〕

(137)　"平成元年 12 月 31 日"と表示するプログラムをつくれ．

(138)　"平成元年 12 月 31 日"のうち元年をとり出して表示するプログラムをつくれ．

(139)　"クイックベーシック　を　学びましょう。"を a$，"始めます。□□"を b$として，a$の最後の 7 文字を b$で入れかえて表示するプログラムをつくれ．

(140)　"たけやがやけた"を後ろから 1 文字ずつ表示するプログラムをつくれ．

(141)　"平成元年 12 月 31 日"の 1 文字ずつの句点コードを表示するプログラムをつくれ．

(142)　"ABCD 大文字 abcd 小文字"のうち半角文字と全角文字を各々とり出して表示するプログラムをつくれ．

(143)　1666 から 2000 までの CHR$の漢字を表示するプログラムをつくれ．

(144)　820 から 852 までの CHR$の文字を表示するプログラムをつくれ．

(145)　"クイックベーシック"を a$とし，a$の JIS$を表示するプログラムをつくれ．

(146)　a$を"クイックベーシック"として，1 文字おきに表示するプログラムをつくれ．

〔解　答〕

(137) **【プログラム例】**

```
PRINT "平成元年１２月３１日"
END
```

【結　果】

```
平成元年１２月３１日
```

(138)【プログラム例】

```
a$ = "平成元年１２月３１日"
b$ = KMID$(a$, 3, 2)
PRINT b$
END
```

【結　果】

```
元年
```

(139)【プログラム例】

```
a$ = "クイックベーシック を 学びましょう。"
b$ = "始めます。       "
KMID$(a$, 13, 7) = b$
PRINT a$
END
```

【結　果】

```
クイックベーシック を 始めます。
```

(140)【プログラム例】

```
a$ = "たけやがやけた"
b = KLEN(a$)
FOR i = b TO 1 STEP -1
    PRINT KMID$(a$, i, 1);
NEXT i
PRINT
END
```

【結　果】

```
たけやがやけた
```

(141)【プログラム例】

```
a$ = "平成元年１２月３１日"
b = KLEN(a$)
FOR i = 1 TO b
    c$ = KMID$(a$, i, 1)
    PRINT KTN$(c$); "  ";
NEXT i
PRINT
END
```

【結　果】

```
4231  3214  2421  3915  0317  0318  2378  0319  0317  3892
```

(142)【プログラム例】

```
a$ = "ABCD大文字abcd小文字"
b$ = KEXT$(a$, 0)
c$ = KEXT$(a$, 1)
PRINT b$
PRINT c$
END
```

【結　果】

```
ABCDabcd
大文字小文字
```

(143)【プログラム例】

```
FOR i = 1666 TO 2000
   a$ = CHR$(i)
   PRINT a$;
NEXT i
END
```

【結　果】

```
亜唖娃阿哀愛挨姶逢葵茜穐悪握渥旭葦芦鯵梓圧斡扱宛姐虻飴絢綾鮎或粟袷安庵按暗案闇鞍
杏以伊位依偉囲夷委威尉惟意慰易椅為畏異移維緯胃萎衣謂違遺医井亥域育郁磯一壱溢逸稲
茨芋鰯允印咽員因姻引飲淫胤蔭院陰隠韻吋右宇烏羽迂雨卯鵜窺丑碓臼渦嘘唄欝蔚鰻姥厩浦
瓜閏噂云運雲荏餌叡営嬰影映曳栄永泳洩瑛盈穎頴英衛詠鋭液疫益駅悦謁越閲榎厭円園堰奄
宴延怨掩援沿演炎焔煙燕猿縁艶苑薗遠鉛鴛塩於汚甥凹央奥往応押旺横欧殴王翁襖鴬鴎黄岡
沖荻億屋憶臆桶牡乙俺卸恩温穏音下化仮何伽価佳加可嘉夏嫁家寡科暇果架歌河火珂禍禾稼
箇花苛茄荷華菓蝦課嘩貨迦過霞蚊俄峨我牙画臥芽蛾賀雅餓駕介会解回塊壊廻快怪悔恢懐戒
拐改魁晦械海灰界皆絵芥蟹開階貝凱劾外咳害崖慨概涯碍蓋街該鎧骸浬馨蛙垣柿蛎鈎劃嚇各
廓拡撹格核殻獲確穫覚角赫較郭閣
```

(144)【プログラム例】

```
FOR i = 820 TO 852
   a$ = CHR$(i)
   PRINT a$;
NEXT i
END
```

【結　果】

```
АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ
```

(145)【プログラム例】

```
a$ = "クィックベーシック"
PRINT JIS$(a$)
END
```

【結　果】

```
252F
```

(146)【プログラム例】

```
a$ = "クィックベーシック"
b = KLEN(a$)
FOR i = 1 TO b STEP 2
    PRINT KMID$(a$, i, 1);
NEXT i
PRINT
END
```

【結　果】

```
クッベシク
```

18章　桁・行の指定

LOCATE

【例題 135】　LOCATE で表示位置を指定

1行1桁から"abc"，2行1桁から"def"，3行10桁から"ghi"5行13桁から"jkl"を表示するプログラムをつくれ.

【解　説】　◎　LOCATE で表示開始位置を決めます.

① LOCATE x，y によって表示開始位置を決めます．x が行，y が桁を示します．下図のように画面左上を1行1桁とします．BASIC では通常，左上を0桁0行とし，x を桁，y を行で表わしますので注意してください.

Quick BASIC

```
(1,1) (1,2) (1,3)………
(2,1)
(3,1)
  ⋮
```

左上を1行1桁とする．行，桁の順に示す.

N_{88} BASIC

```
(0,0) (1,0) (2,0)………
(0,1)
(0,2)
  ⋮
```

左上を0桁0行とする．桁，行の順に示す.

② LOCATE　1，1 では1行1桁にカーソルを移します．続いて，PRINT "abc"により，1行1桁から"abc"を表示します.

③ LOCATE　2，1 では2行1桁にカーソルを移します．続いて，PRINT "def"で，2行1桁から"def"を表示します.

④ LOCATE，10 では行を変更せず10桁へカーソルを移します．すなわち項③の PRINT "def"で改行されていますから，実際は3行目，10桁にカーソルがきます．そこから"ghi"を表示します.

⑤ LOCATE　5 では5行目の桁は変更しない位置へカーソルを移します．実際は3行目，10桁目から"ghi"；を表示していますのでカーソルは3行目13桁目にあります．したがって，桁は同じですから5行目の13桁目にカーソルを移します．そこから PRINT "jkl"で表示をします.

【プログラム例】

```
LOCATE 1, 1 ─┐ 1行1桁から abc を表示
PRINT "abc" ─┘
LOCATE 2, 1 ─┐ 2行1桁から def を表示
PRINT "def" ─┘
LOCATE , 10 ─┐ 3行10桁から ghi を表示
PRINT "ghi"; ┘
```

```
LOCATE 5 ─────┐─5行13桁からjklを表示
PRINT "jkl"───┘
END
```

【結　果】

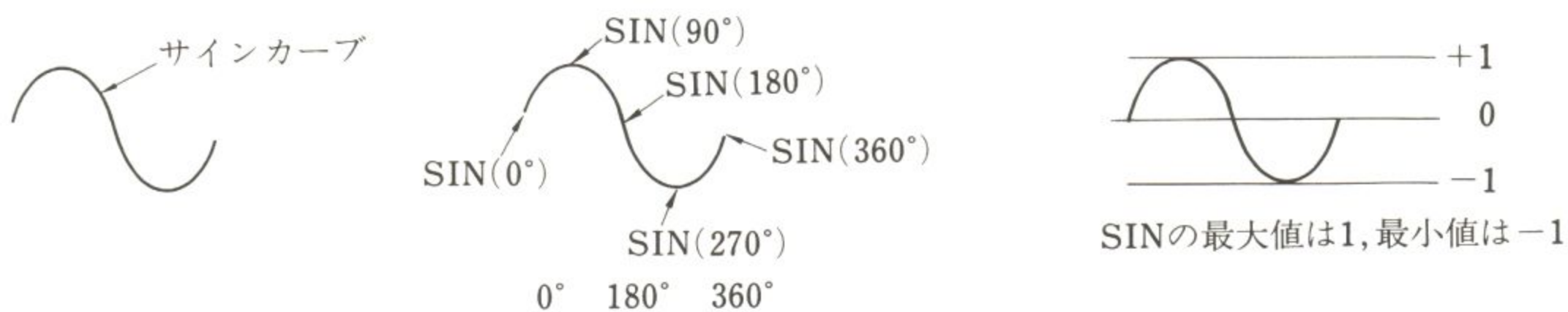
```
abc
def
              ghi

                   jkl
```

【例題 136】　サインカーブを描く

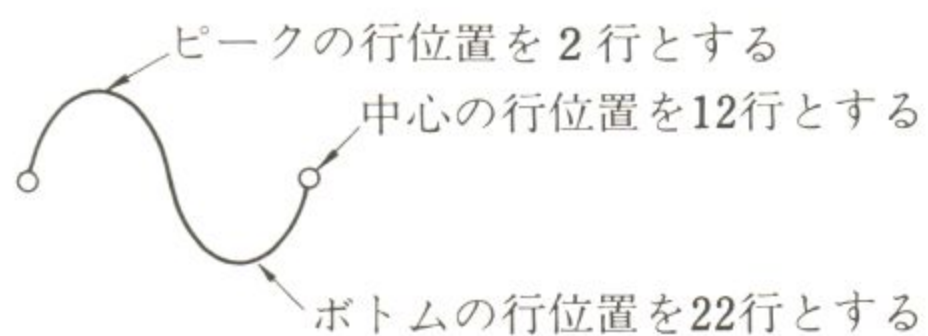
"0"を使ってサインカーブをかくプログラムをつくれ.

【解　説】　◎　LOCATEを使ってサインカーブを描きます.

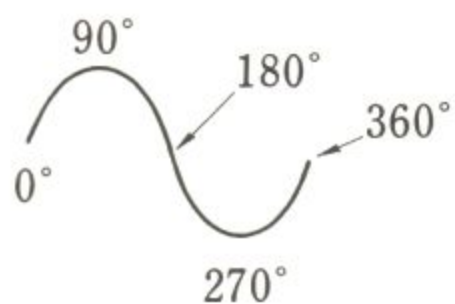
① サインカーブは0度から360度までのSINの値をプロットしたものです.

サインカーブ
SIN(90°)
SIN(180°)
SIN(0°)
SIN(360°)
SIN(270°)
0°　180°　360°
+1
0
−1
SINの最大値は1, 最小値は−1

② キャラクタ"0"をLOCATEを使って表示します. 次のように考えます.

〔行位置の決め方〕

ピークの行位置を2行とする
中心の行位置を12行とする
ボトムの行位置を22行とする

中心を12行, SINの最大値1のとき2行目
SINの最小値-1のとき22行目とする.
したがって, 行位置は12-10SIN(0〜360°)となります.
0〜360°は適当な間隔とします. 桁位置の決定のところで説明します.

〔桁位置の決め方〕

90°
180°
360°
0°
270°

0°のときの桁位置を4桁, 90°のときの桁位置を22桁
180°の桁位置を40桁, 270°の桁位置を58桁
360°の桁位置を76桁とします.
表示は0°, 5°, 10°, 15°, …355°, 360°の場合の桁位置のときとします.
したがって, 4桁目, 5桁目, 6桁目, …76桁目に表示されます.

③ したがって, FOR i=0　TO　360　STEP　5のくり返しで, i度のときの行位置と桁位置を求めると次のようになります.

12-10*SIN(i/180*3.14159)　　……行位置

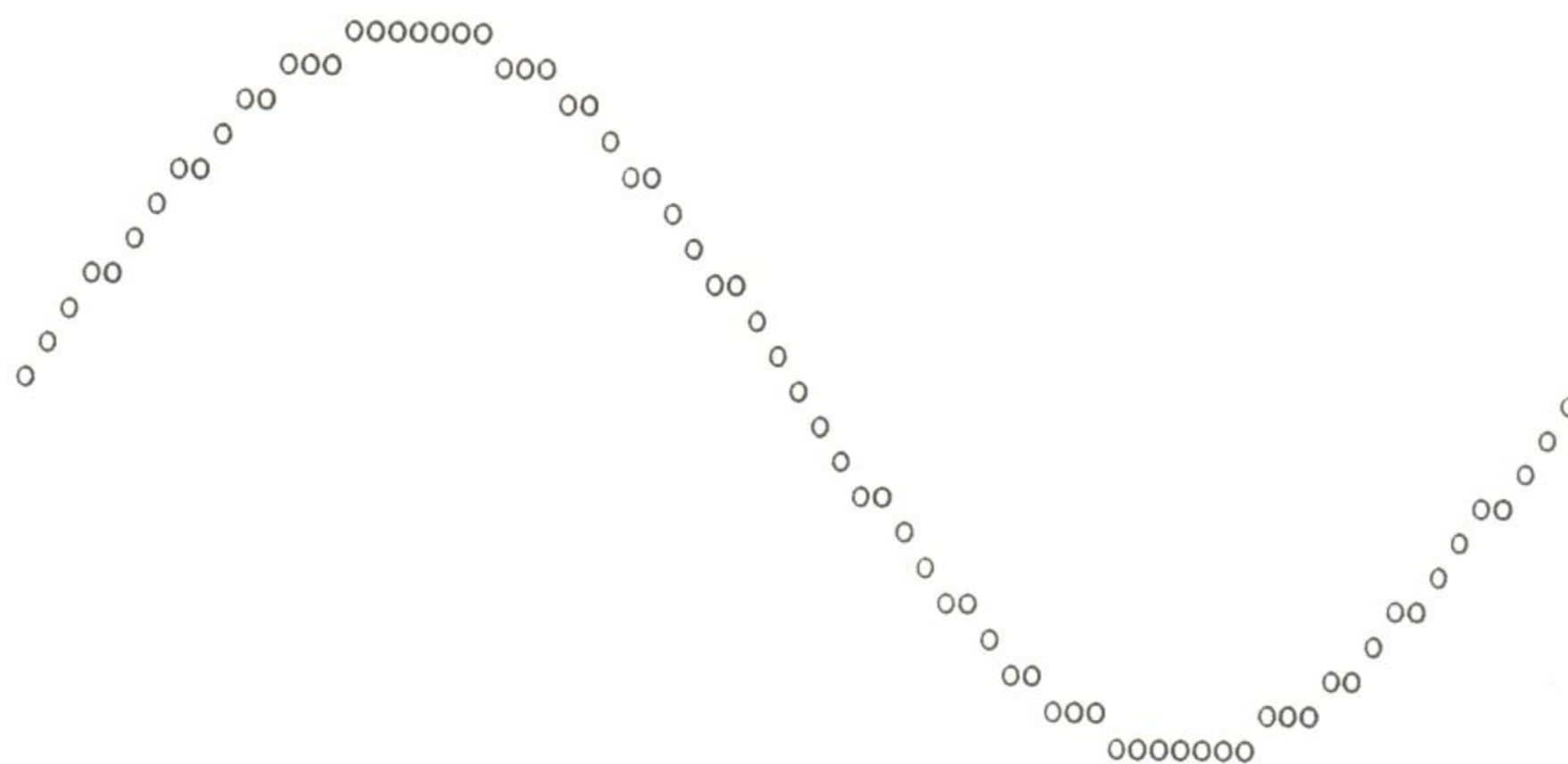
(i+20)/5　　……桁位置

④ したがって, LOCATE i, (i+20)/5によって"0"を表示する位置を決めます.

【プログラム例】

```
FOR i = 0 TO 360 STEP 5 ――――――――――― 0〜360°まで5°ごとにくり返し
    j = 12 - 10 * SIN(i / 180 * 3.14159)―――― 行の計算
    LOCATE j, (i + 20) / 5 ――――――――――― j行，(i+20)/5桁を指定
    PRINT "O"―――――――――――――――――― Oの表示
NEXT i
END
```

【結　果】

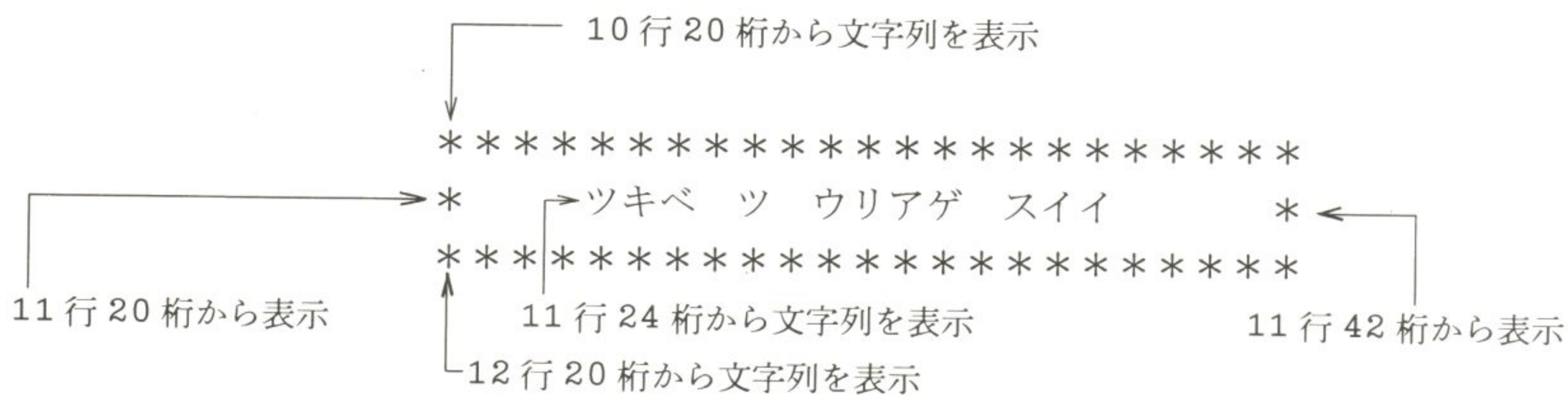
```
                 ooooooo
              ooo       ooo
            oo             oo
           o                 o
         oo                   oo
        o                       o
       o                         o
     oo                           oo
    o                               o
   o                                 o
  o                                   o
 o                                     o
                                        o
                                        o
                                         oo
                                           o
                                            o
                                             oo
                                               o
                                                oo
                                                  ooo
                                                     ooooooo
```

【例題 137】　標題を表示

結果の標題を左上を10行20桁として表示するプログラムをつくれ．

【解　説】　◎　標題を指定位置へ表示します．

①　"*"を23個並べた文字列をa$，"*"1つをb$，標題の文字列をc$に代入します．文字列は全体で15桁ですから，前後に3桁ずつ空白をつけ，またその前後に"*"を1つずつつけます．したがって，全体は23桁です．したがって，上下の行に表示する"*"の文字列は23個の"*"を持たせます．

②　次のようにLOCATE文で位置を指定して表示します．

```
                      ┌── 10行20桁から文字列を表示
                      ↓
                      ***********************
  ┌─────────────────→ *   →ツキベ ツ ウリアゲ スイイ   * ←─┐
  │                   ***|*******************           │
11行20桁から表示      ↑  11行24桁から文字列を表示   11行42桁から表示
                      └12行20桁から文字列を表示
```

【プログラム例】

```
READ a$, b$, c$ ――――――――――― データ文の文字列を代入
LOCATE 10, 20 ―――――――――――┐ a$の表示
PRINT a$ ―――――――――――――――┘
LOCATE 11, 20 ―――――――――――┐ b$の表示
PRINT b$ ―――――――――――――――┘
```

```
LOCATE 11, 24 ──────┐ c$の表示
PRINT c$ ───────────┘
LOCATE 11, 42 ──────┐ b$の表示
PRINT b$ ───────────┘
LOCATE 12, 20 ──────┐ a$の表示
PRINT a$ ───────────┘
END
DATA "************************"
DATA "*"
DATA "ﾂｷﾍﾞﾂ ｳﾘｱｹﾞ ｽｲｲ"
```

【結　果】

```
************************
*    ﾂｷﾍﾞﾂ ｳﾘｱｹﾞ ｽｲｲ    *
************************
```

【例題 138】　位置指定で入力

2 行 5 桁目から文字を入力し，4 行 5 桁目から表示するプログラムをつくれ.

【解　説】　◎　LOCATE は入力位置の指定もできます.

① LOCATE　2, 5 の後 INPUT x$とするとカーソルを 2 行 5 桁へ移動し，そこから x$の入力を行います. ただし，INPUT 文の?の位置が 5 桁目です.

?　────　LOCATE　2, 5 により INPUT 文の?は 2 行 5 桁目に表示される

【プログラム例】

```
LOCATE 2, 5 ──┐ 2 行 5 桁から x$の入力
INPUT x$ ─────┘
LOCATE 4, 5 ──┐ 4 行 5 桁から表示
PRINT x$ ─────┘
END
```

【結　果】

```
?                    入力待ち画面




? 123456             123456 を入力の例

123456
```

【例題 139】　文字の移動

A を画面左から右へ動かすプログラムをつくれ.

【解　説】　◎　文字を画面左から右へ動かします.

①　文字の移動は次のようにします.

```
A   A   A   A  ..........................  A     Aが左から右へ動く.
↑└─┘↑└─┘↑└─┘↑     └──────────────────────┘↑     同じ行の桁を1から80までにすると1桁目か
│                                                 ら80桁目まで動く.
Aを表示
次にスペースを表示
```

【プログラム例】

```
FOR i = 1 TO 80―――――iを1から80まで, iは桁を示す
    LOCATE 3, i――――┐3行i桁にAを表示
    PRINT "A"―――――┘
    LOCATE 3, i――――┐3行i桁にスペースを表示
    PRINT " "――――――┘
NEXT i
END
```

【結　果】

A　　　　　　Aが左から右へ動く

〔演習問題〕

(147)　下のデータを1行目～9行目の5桁目から1つずつ表示するプログラムをつくれ.

データ：PC-9801, PC-8801, PC-98DO

IBM-XT, IBM-PS2, IBM-AT

FM-towns, Canon-AXi, Panacom-M530

(148)　Aを結果のように表示するプログラムをつくれ. これは"A"を1行1桁, 2行2桁, 3行3桁, …15行15桁に表示したものである.

(149)　"O"を使ってコサインカーブを表示するプログラムをつくれ.

〔解　答〕

(147)【プログラム例】

```
FOR i = 1 TO 9―――――――――――――――――9回くり返し
    READ a$ ――――――――――――――――――文字列を読む
    LOCATE i, 5 ――――――――――――――┐i行5桁から表示
    PRINT a$ ―――――――――――――――――┘
NEXT i
END
DATA PC-9801,PC-8801,PC-98DO
DATA IBM-XT,IBM-PS2,IBM-AT
DATA FM-towns,Canon-AXi,Panacom-M530
```

【結　果】

```
PC-9801
PC-8801
PC-98DO
IBM-XT
IBM-PS2
IBM-AT
FM-towns
Canon-AXi
Panacom-M530
```

(148)【プログラム例】

```
FOR i = 1 TO 15
    LOCATE i, i      ─┐ i 行 i 桁から"A"を表示
    PRINT "A"        ─┘
NEXT i
END
```

【結　果】

```
A
 A
  A
   A
    A
     A
      A
       A
        A
         A
          A
           A
            A
             A
              A
```

(149)【プログラム例】

```
FOR i = 0 TO 360 STEP 5
    j = 12 - 10 * COS(i / 180 * 3.14159)  ── 行位置を求める
    LOCATE j, (i + 20) / 5                ─┐ j 行, (i+20)/5 桁に o を表示
    PRINT "O"                              │
NEXT i                                    ─┘
END
```

【結　果】

```
oooo                                                          oooo
    ooo                                                    ooo
       oo                                               oo
         o                                             o
          oo                                         oo
            o                                       o
             o                                     o
              oo                                 oo
                o                               o
                 o                             o
                  o                           o
                   o                         o
                    o                       o
                     oo                   oo
                       o                 o
                        o               o
                         oo           oo
                           o         o
                            oo     oo
                              ooo ooo
                                 ooooooo
```

19章　タイマー割り込み・キー割り込み・サブルーチンへ分岐

TIMER ON/OFF/STOP，ON TIMER GOSUB～RETURN
KEY( ) ON/OFF/STOP，ON KEY( )GOSUB～RETURN

【例題140】　タイマー割り込み

FOR i=1　TO　30000:NEXT i をくり返す中で，1秒ごとに割り込みでサブプログラム count へ分岐し，カウンタを1ずつ増やし表示するプログラムをつくれ.

【解　説】　◎　タイマー割り込みをします.

① 次の形でくり返しの途中で1秒ごとに割り込みがかかり，サブプログラム count へ分岐します.

```
TIMER ON                          割り込みをオンとする
 |
ON TIMER(1)  GOSUB count
         └──── 1秒ごとに割り込みサブプログラム count へ分岐
FOR i=1  TO  30000:NEXT i
```

② 割り込みの開始は TIMER ON です．割り込みの停止は TIMER OFF，割り込みの中断は TIMER STOP です.

③ サブルーチンへの分岐は GOSUB プログラム名です．プログラム名はラベルのように使います．サブルーチンの終了は RETURN です.

```
GOSUB count ──┐
～            │   count の名前のサブルーチンへ分岐
count:   ←────┘
～
RETURN            RETURN で GOSUB の次へ戻る
```

④ このプログラムはほとんどが FOR i=1　TO　30000:NEXT i の処理に時間を使います．その間1秒ごとに割り込みが入り，サブプログラム count へ分岐して，c に1を加え表示していきます．このプログラムの処理に約16秒かかることがわかります.

【プログラム例】

```
TIMER ON ──────────────── 割り込み開始
ON TIMER(1) GOSUB count ──── 1秒ごとに割り込みでサブプログラム count へ分岐
```

```
FOR i = 1 TO 30000: NEXT i ──── くり返し
PRINT
END
count: ──────────────┐ サブプログラム
    c = c + 1        │ c に 1 を加えて表示
    PRINT c;         │
RETURN ──────────────┘
```

【結　果】

```
1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16
```

【例題 141】　キー割り込み

ファンクションキー1，2，3 がおされたら，割り込みで"キー1　ガ　オサレマシタ"，"キー2　ガ　オサレマシタ"，"キー3　ガ　オサレマシタ"と表示して終了するプログラムをつくれ．ただし，キー3 がおされたら表示の後終了するものとする．

【解　説】　◎　ファンクションキーによる割り込みをします．

① ファンクションキー1～10 を押すことで割り込みをさせることができます．

② KEY(1)　ON　と ON KEY(1)　GOSUB a によってファンクションキー1 がおされると割り込みでサブプログラム a へ分岐します．

③ KEY(1)の 1 の部分が 1 から 10 を使いファンクションキー1～10 を示します．

④ KEY()　OFF は割り込みの禁止，KEY()　STOP は割り込みを中断します．

⑤ KEY 割り込みが ON のときはファンクションキーに割りあてられているファンクションは無効です．

【プログラム例】

```
KEY(1) ON ─────────────┐ ファンクションキー1～3 の割り込みをオン
KEY(2) ON              │
KEY(3) ON ─────────────┘
ON KEY(1) GOSUB a ─────┐ ファンクションキー1～3 からおされると各々
ON KEY(2) GOSUB b      │ サブプログラム a，b，c へ分岐
ON KEY(3) GOSUB c ─────┘
WHILE 1 ───────────────┐ 無限ループ
    z = z              │
WEND ──────────────────┘
e:
END
a: ────────────────────┐ サブプログラム a
    PRINT "ｷｰ1 ｶﾞ ｵｻﾚﾏｼﾀ"  │
RETURN ────────────────┘
b: ────────────────────┐ サブプログラム b
    PRINT "ｷｰ2 ｶﾞ ｵｻﾚﾏｼﾀ"  │
RETURN ────────────────┘
c: ────────────────────┐ サブプログラム c
    PRINT "ｷｰ3 ｶﾞ ｵｻﾚﾏｼﾀ"  │
RETURN e ──────────────┘
```

【結　果】

```
キー1 ガ オサレマシタ
キー2 ガオサレマシタ
キー2 ガオサレマシタ
キー1 ガ オサレマシタ
キー3 ガ オサレマシタ
```

ファンクションキー
1, 2, 2, 1, 3 がおされた例

【例題 142】　矢印キーによる割り込み

矢印キーがおされたらその方向の矢印を表示するプログラムをつくれ.

【解　説】　◎　矢印キーによる割り込みをします.

① [↑] キーは 11, [←] キーは 12, [→] キーは 13, [↓] キーは 14 によって割り込みができます. 使い方は【例題 141】と同じです.

【プログラム例】

```
KEY(11) ON ─────┐ キー11〜14 の割り込みオン
KEY(12) ON      │ 11 は[↑], 12 は[←], 13 は[→], 14 は[↓] キー
KEY(13) ON      │
KEY(14) ON ─────┘
ON KEY(11) GOSUB a ─┐ キー11〜14 による割り込みによってサブプログラム a〜d へ分岐
ON KEY(12) GOSUB b  │
ON KEY(13) GOSUB c  │
ON KEY(14) GOSUB d ─┘
WHILE 1 ────────┐ 無限ループ
    z = z ──────┘
WEND
e:
END
a: ─────────────┐ サブプログラム a
    PRINT "↑"   │
RETURN e ───────┘
b: ─────────────┐ サブプログラム b
    PRINT "←"   │
RETURN e ───────┘
c: ─────────────┐ サブプログラム c
    PRINT "→"   │
RETURN e ───────┘
d: ─────────────┐ サブプログラム d
    PRINT " ↓"  │
RETURN e ───────┘
```

【結　果】

```
↑
→
 ↓
←
 ↓
```

[↑], [→], [↓], [←], [↓] を入力の例

【例題 143】　ユーザー定義キー割り込み

A キー，% キー，CTRL キーと C キーがおされたら各々A，%，end と表示するプログラムをつくれ．

【解　説】　◎　ユーザー定義キーを使ってキー割り込みをします．

① キー番号 15 から 25 はユーザーが定義して使います．

② 定義の仕方は次のとおりです．

```
KEY n ,   CHR$(&Hx)+CHR$(&Hy)
```

- n：15 から 25 まで
- &Hx：&H00 はキーボードフラッグなし，&H01 は SHIFT キー，&H02 は CAPS キー，&H04 は カナ キー，&H08 は GRPH キー，&H10 は CTRL キー
- &Hy：キーボードスキャンコードで，ESC が 00，1! が 01，2" が 02…，CTRL が 74 のようになっています．

③ たとえば，次のようになります．

```
KEY  15,CHR$(&H00)+CHR$(&H1D)     A キーを 15 に割りあて
KEY  16,CHR$(&H01)+CHR$(&H05)     SHIFT キーと 5% キーを 16 に割りあて
KEY  17,CHR$(&H10)+CHR$(&H2B)     CTRL キーと C キーを 17 に割りあて
```

【プログラム例】

```
KEY 15, CHR$(&H0) + CHR$(&H1D)  ─┐キー15～17 の定義
KEY 16, CHR$(&H1) + CHR$(&H5)    │
KEY 17, CHR$(&H10) + CHR$(&H2B) ─┘
KEY(15) ON  ─┐キー15～17 のオン
KEY(16) ON   │
KEY(17) ON  ─┘
ON KEY(15) GOSUB a ─┐キー割り込み
ON KEY(16) GOSUB b  │
ON KEY(17) GOSUB c ─┘
WHILE 1 ─┐無限ループ
   z = z  │
WEND    ─┘
e:
END
a: ─┐サブプログラム a
   PRINT "A"  │
RETURN e ─┘
b: ─┐サブプログラム b
   PRINT "%"  │
              │
RETURN e ─┘
```

```
c: ─────────────────────── サブプログラム c
    PRINT "end"
RETURN e ──────────────────
```

【結　果】

```
A          Ⓐ, %, CTRL + Ⓒ を入力の例
%
end
```

【例題 144】　割り込みを使ってプログラムの中断を防止

STOP キー, ESC キー, CTRL + Ⓒ キーを各々15, 16, 17 にキー定義して, 1 から 1000 までを表示するプログラムの中でこれらがおされてもプログラムが中断しないプログラムをつくれ.

【解　説】　◎　キー割り込みを利用してプログラムの中断を防ぎます.

① STOP キーをプログラム実行中におすとプログラムが中断します. これを防ぐためにはキー割り込みを利用して, STOP キーがおされたら, サブプログラムへ分岐して, 単に RETURN すれば, プログラムは中断しません.

② 同様に ESC キー, CTRL キー + Ⓒ キーを定義してキー割り込みを利用してプログラムの中断を防ぎます.

【プログラム例】

```
KEY 15, CHR$(&H0) + CHR$(&H60) ──  STOP キーを 15, ESC キーを 16, CTRL
KEY 16, CHR$(&H0) + CHR$(&H0)      キー+ Ⓒ キーを 17 で定義
KEY 17, CHR$(&H10) + CHR$(&H2B) ─
KEY(15) ON ──────────────────── キー15～17 のオン
KEY(16) ON
KEY(17) ON ──────────────────
ON KEY(15) GOSUB a ──────────── キー割り込み
ON KEY(16) GOSUB b
ON KEY(17) GOSUB c ──────────
FOR i = 1 TO 1000 ───────────── 1～1000 の表示
  PRINT i;
NEXT i ──────────────────────
PRINT
END
a: ──────────────────────────── サブプログラム a～c, いずれも単に RETURN のみ,
    RETURN                       プログラムの中断を防ぐ
b:
    RETURN
c:
    RETURN ──────────────────
```

【結　果】　前半省略　(ESC キー, STOP キー, CTRL + Ⓒ キーを途中でおす)

```
8  919  920  921  922  923  924  925  926  927  928  929  930  931  932  933  93
4  935  936  937  938  939  940  941  942  943  944  945  946  947  948  949  95
0  951  952  953  954  955  956  957  958  959  960  961  962  963  964  965  96
6  967  968  969  970  971  972  973  974  975  976  977  978  979  980  981  98
2  983  984  985  986  987  988  989  990  991  992  993  994  995  996  997  99
8  999  1000
```

〔演習問題〕

(150)　10 秒たったら，"10 ビョウ　ケイカ"と表示して終わるプログラムをつくれ．

(151)　Ⓐ，Ⓑ，Ⓒ キーがおされたらその数を数えて表示するプログラムをつくれ．ただし，どれかが 5 回となったら表示して終了するものとする．

〔解　答〕

(150)【プログラム例】

```
TIMER ON ―――――――――――――――― タイマ割り込み開始
ON TIMER(10) GOSUB count ――― 10 秒たったら count へ分岐
WHILE 1 ―――――――――――――――┐ 無限ループ
    z = z                   │
WEND ――――――――――――――――――┘
e:
END
count: ―――――――――――――――――┐ サブプログラム count
    PRINT "10 ビョウ ケイカ"  │
RETURN e ――――――――――――――――┘
```

【結　果】

```
10 ビョウ ケイカ
```

(151)【プログラム例】

```
KEY 15, CHR$(&H0) + CHR$(&H1D) ――┐ Ⓐ，Ⓑ，Ⓒ キーを定義
KEY 16, CHR$(&H0) + CHR$(&H2D)   │
KEY 17, CHR$(&H0) + CHR$(&H2B) ――┘
KEY(15) ON ――――――――――――――――――――┐ キー15～16 オン
KEY(16) ON                       │
KEY(17) ON ――――――――――――――――――――┘
ON KEY(15) GOSUB a ――――――――――――┐ キー割り込み
ON KEY(16) GOSUB b               │
ON KEY(17) GOSUB c ――――――――――――┘
WHILE 1 ―――――――――――――――――――――――┐ 無限ループ
   z = z                         │
WEND ――――――――――――――――――――――――――┘
e:
PRINT "A="; a; "カイ " ―――――――――┐ 表示
PRINT "B="; b; "カイ "            │
PRINT "C="; c; "カイ " ―――――――――┘
END
a: ――――――――――――――――――――――――――――┐ サブプログラム a
    a = a + 1                      │ Ⓐ の回数のカウント
    IF a <= 4 THEN RETURN ELSE RETURN e ┘
```

```
b: ─────────────────────────────────── サブプログラムb
    b = b + 1                           Ⓑの回数のカウント
    IF b <= 4 THEN RETURN ELSE RETURN e
c: ─────────────────────────────────── サブプログラムc
    c = c + 1                           Ⓒの回数のカウント
    IF c <= 4 THEN RETURN ELSE RETURN e
```

【結　果】

```
A= 3 カイ          (Ⓑ Ⓐ Ⓑ Ⓑ Ⓐ Ⓒ Ⓒ Ⓐ Ⓒ Ⓑ Ⓑ の例)
B= 5 カイ
C= 3 カイ
```

20章　シーケンシャルファイル

OPEN～FOR OUTPUT, OPEN～FOR INPUT, OPEN～FOR APPEND,
WRITE#, INPUT#, PRINT#, CLOSE#, EOF

【例題 145】　シーケンシャルファイルの書き込み

下のデータをシーケンシャルファイル meibo.dat にかくプログラムをつくれ.

データ：オオヤマタロウ, 38, 1570, ウエダヨシオ, 41, 1300
　　　　クリタサブロウ, 25, 820, トクガワヨシオ, 52, 2300
　　　　カワカミタロウ, 21, 420, アマダヨシミツ, 37, 1210
　　　　（名前，年齢，年収の順のデータとする）

【解　説】 ◎ シーケンシャルファイルをつくり書き込みをします.

① シーケンシャルファイルはテープと同じように前から順にデータを書いていきます. 途中から書くことはできません.

② まずファイルをオープンします. ファイルのオープンは今後使用するファイルを明確にし，それに番号を与えて，今後その番号で識別していきます. シーケンシャルファイルの書き込みは OUTPUT を指定します.

```
OPEN     "meibo.dat" FOR OUTPUT AS#1
↑          ↑          ↑          ↑
ファイルの  ファイル名   書き込み用   オープン番号
オープン
```

この中でユーザーが変えられるのはファイル名の meibo.dat の部分とオープン番号の 1 のみです. 1 から 255 が使えます. あとは書式のとおりとします.

③ 配列 a$ と x% に読み込まれたデータを次の形でファイルへ書き込みます. 書き込みは WRITE# を使います. ファイルの識別をオープン番号 1 で識別します.

```
WRITE   #1  a$(i),x%(i,1),x%(i,2)
↑       ↑   ↑        ↑        ↑
ファイルへ 識別  名前     年齢     年収
の書き込み 番号1
```

④ 書き込みが終ったらオープン番号と同じ番号を指定してファイルをクローズします.

```
CLOSE  #1
       ↑
ファイルの └── 識別番号
クローズ       #1 を省略するとすべてのファイルがクローズされます.
```

CLOSE がなくても END などによって自動的にクローズされます.

⑤ シーケンシャルファイルでは本例のように，文字データ，整数型データ，整数型データを１組として，この順でくり返されます．このとき，文字データの長さが同じでなくてもよいところに大きな特長があります．

したがって，このファイルが何バイト使用しているのかデータの数だけではわかりません．

⑥ FOR OUTPUT を指定してファイルをオープンすると，同じ名前のファイルがなければ作成し，すでにあるときはそのデータをすべてなくしてしまいますから注意してください．

【プログラム例】

```
DIM a$(6), x%(6, 2) ──────────────── 配列の宣言
FOR i = 0 TO 5 ──────────────┐────── データの読み込み
   READ a$(i), x%(i, 1), x%(i, 2)    │
NEXT i ──────────────────────┘
                                      ファイルのオープン
OPEN "b:meibo.dat" FOR OUTPUT AS #1 ─┐── ファイルへデータの書き込み
FOR i = 0 TO 5                        │
   WRITE #1, a$(i), x%(i, 1), x%(i, 2)│
NEXT i ───────────────────────────────┘
CLOSE #1 ───────────────────────────── ファイルのクローズ
END
DATA ｵｵﾔﾏﾀﾛｳ,38,1570,ｳｴﾀﾞ ﾖｼｵ,41,1300
DATA ｸﾘﾀｻﾌﾞ ﾛｳ,25,820,ﾄｸｶﾞ ﾜﾖｼｵ,52,2300
DATA ｶﾜｶﾐﾀﾛｳ,21,420,ｱﾏﾀﾞ ﾖｼﾐﾂ,37,1210
```

【結　果】　データがディスクにかき込まれます．

MS-DOS の type コマンドでファイルの内容をみると次のようになっています．

```
type meibo.dat
"ｵｵﾔﾏﾀﾛｳ",38,1570
"ｳｴﾀﾞﾖｼｵ",41,1300
"ｸﾘﾀｻﾌﾞﾛｳ",25,820
"ﾄｸｶﾞﾜﾖｼｵ",52,2300
"ｶﾜｶﾐﾀﾛｳ",21,420
"ｱﾏﾀﾞﾖｼﾐﾂ",37,1210
```

【例題 146】　シーケンシャルファイルの読みだし

【例題 145】で書き込んだシーケンシャルファイル meibo.dat のデータを読み出して表示するプログラムをつくれ．

【解　説】　◎　シーケンシャルファイルを読み出します．

① シーケンシャルファイルはテープと同様に前から順にしか読み出せません．

② シーケンシャルファイルの読み出しは，書き込みと同様にまずファイルをオープンします．このとき，INPUT を指定します．

OPEN	"meibo.dat"	FOR INPUT	AS #2
ファイルのオープン	ファイル名	読み出しの指定	オープン番号

③ 読み出しは次のようにします．読み出しは INPUT# を使います．

```
INPUT  #2,  a$,  x%,  y%
```

読み出された最初の文字列がa$に，次の2つの整数がx%とy%に代入されます.

④　読み出しがすんだら，ファイルをクローズします.

```
CLOSE  #2
       ↑
クローズ └──── 識別番号
```

【プログラム例】

```
OPEN "b:meibo.dat" FOR INPUT AS #2 ───ファイルのオープン
FOR i = 1 TO 6 ──────────────┐─データの読み出しと表示
    INPUT #2, a$, x%, y%       │
    PRINT a$; x%; y%           │
NEXT i ────────────────────┘
CLOSE #2 ──────────────────ファイルのクローズ
END
```

【結　果】

```
ｵｵﾔﾏﾀﾛｳ 38   1570
ｳｴﾀﾞﾖｼｵ 41   1300
ｸﾘﾀｻﾌﾞﾛｳ 25   820
ﾄｸｶﾞﾜﾖｼｵ 52   2300
ｶﾜｶﾐﾀﾛｳ 21   420
ｱﾏﾀﾞﾖｼﾐﾂ 37   1210
```

【例題 147】　シーケンシャルファイルにデータを追加

【例題 145】のファイル meibo.dat に追加して下のデータを書き込むプログラムをつくれ.

データ：カドカワヒロシ，35，2100，オオシタヤスオ，43，990

【解　説】　◎　シーケンシャルファイルにデータを追加書き込みします.

①　シーケンシャルファイルにデータを追加書き込みするときは，ファイルをオープンするときに FOR APPEND を使います.

```
OPEN   "meibo.dat"  FOR APPEND AS #1
↑       ↑            ↑              └┐
ファイルの ファイル名  追加書き込みを指定  オープン番号
オープン
```

②　書き込み，クローズは書き込みを FOR OUTPUT 指定で行ったのと同じです.

③　APPEND を使って追加書き込みをするときはファイル名が同じものがあれば，追加で書き込み，すなわち前のデータをこわさないようにオープンします．同じファイル名がないときはその名前でファイルをつくります．つまり OUTPUT を使ったときと同じです．したがって，誤ってファイルをこわさないように OUTPUT より APPEND を使った方が無難です.

【プログラム例】

```
DIM a$(2), x%(2, 2) ──────────────配列の宣言
FOR i = 0 TO 1 ─────────────────┐─データの読み込み
    READ a$(i), x%(i, 1), x%(i, 2)  │
NEXT i ─────────────────────────┘
```

```
OPEN "b:meibo.dat" FOR APPEND AS #3 ―――― ファイルのオープン
FOR i = 0 TO 1 ――――――――――――――― データの追加書き込み
    WRITE #3, a$(i), x%(i, 1), x%(i, 2)
NEXT i
CLOSE #3 ―――――――――――――――――― ファイルのクローズ
END
DATA ｶﾄﾞｶﾜﾋﾛｼ,35,2100,ｵｵｼﾀﾔｽｵ,43,990
```

【結　果】　データがディスクに書き込まれます.

MS-DOS の type コマンドでファイルの内容を表示すると次のようになります.

```
"ｵｵﾔﾏﾀﾛｳ",38,1570
"ｳｴﾀﾞﾖｼｵ",41,1300
"ｸﾘﾀｻﾌﾞﾛｳ",25,820
"ﾄｸｶﾞﾜﾖｼｵ",52,2300
"ｶﾜｶﾐﾀﾛｳ",21,420
"ｱﾏﾀﾞﾖｼﾐﾂ",37,1210
"ｶﾄﾞｶﾜﾋﾛｼ",35,2100
"ｵｵｼﾀﾔｽｵ",43,990
```

【例題 148】　◎シーケンシャルファイルの EOF を使った読み出し

【例題 145】と【例題 147】を実行した後のファイル meibo.dat のデータを読み出して表示するプログラムをつくれ.

【解　説】　◎　シーケンシャルファイルのデータを EOF を利用して読み出します.

①　一般にシーケンシャルファイルにどれだけデータが入っているかをおぼえていることはありません. そこで, ファイルを読むとき FOR～NEXT 文で読もうと思ってもくり返しの数がわかりません. そこで EOF を利用します.

②　EOF はファイルにデータがあると 0（偽）, なくなると-1（真）を持ちます. これを利用して, ファイルにデータがある間読み出します.

```
                ┌――――― 識別番号
                ↓
DO UNTIL EOF(1)              オープン番号 1 のファイルのデータがなくなると
   INPUT #1,a$,x%, y%        EOF(1)は－1 となる
LOOP
```

【プログラム例】

```
OPEN "b:meibo.dat" FOR INPUT AS #1 ―― ファイルのオープン
DO UNTIL EOF(1) ――――――――――――― データの読み出しと表示
    INPUT #1, a$, x%, y%
    PRINT a$; x%; y%
LOOP
CLOSE #1 ――――――――――――――――― ファイルのクローズ
END
```

【結　果】

```
ｵｵﾔﾏﾀﾛｳ 38   1570
ｳｴﾀﾞﾖｼｵ 41   1300
ｸﾘﾀｻﾌﾞﾛｳ 25   820
ﾄｸｶﾞﾜﾖｼｵ 52   2300
ｶﾜｶﾐﾀﾛｳ 21   420
ｱﾏﾀﾞﾖｼﾐﾂ 37   1210
ｶﾄﾞｶﾜﾋﾛｼ 35   2100
ｵｵｼﾀﾔｽｵ 43   990
```

【例題 149】　シーケンシャルファイルのデータの変更

【例題 148】のファイル meibo.dat の第 3 組目のデータを下のデータで変更するプログラムをつくれ.

　　データ;タジマコウジ, 55, 1920

【解　説】　◎　シーケンシャルファイルの 1 部のデータを変更します.

① シーケンシャルファイルは前から順に書き，前から順に読み出しますから途中で読み出したり，途中で書いたりすることはできませんでした. Quick BASIC では seek を使ってできるようになっています. この例はⅡ編 23 章で扱います. 本例はまず従来の BASIC と同じ方法によってプログラムします.

② 全データを【例題 148】の方法で読み出し配列へ格納します.

③ 次に 3 番目を指定して配列データを入れかえます.

④ 全データをファイルへ書き込みます.

⑤ 読み出すとき EOF( )を使うと終りを検出できますが i=i+1 によってデータ数を求めておきます. 次に書き込むとき，i 個分をかけばよいことになります.

【プログラム例】

```
DIM aa$(100), xx%(100, 2)                    ─ 配列の宣言
i = 0                                        ─ データ組数を数えるカウンタ
OPEN "b:meibo.dat" FOR INPUT AS #2           ─ ファイルのオープン
DO UNTIL EOF(2)                              ┐ データの読み出し配列へ代入
    i = i + 1                                │
    INPUT #2, a$, x%, y%                     │
    aa$(i) = a$                              │
    xx%(i, 1) = x%                           │
    xx%(i, 2) = y%                           │
LOOP                                         ┘
CLOSE #2                                     ─ ファイルのクローズ
aa$(3) = "ﾀｼﾞﾏｺｳｼﾞ"                          ┐ 新しいデータを代入
xx%(3, 1) = 55                               │
xx%(3, 2) = 1920                             ┘
OPEN "b:meibo.dat" FOR OUTPUT AS #5          ─ ファイルのオープン
FOR j = 1 TO i                               ┐ データの書き込み
    WRITE #5, aa$(j), xx%(j, 1), xx%(j, 2)   │
NEXT j                                       ┘
CLOSE #5                                     ─ ファイルのクローズ
END
```

【結　果】 (ファイルの内容)

```
オオヤマタロウ 38   1570
ウエダ゛ヨシオ 41   1300
タシ゛マコウシ゛ 55   1920
トクカ゛ワヨシオ 52   2300
カワカミタロウ 21   420
アマダ゛ヨシミツ 37   1210
カト゛カワヒロシ 35   2100
オオシタヤスオ 43   990
```

【例題 150】　PRINT #を使ってファイルへデータの書き込み

【例題 145】のデータをファイル meibo2.dat に PRINT #文を使ってかくプログラムをつくれ.

【解　説】 ◎ PRINT #を使ってファイルへデータをかきます.

① WRITE #のかわりに PRINT #を使ってデータをファイルへかきます.

② PRINT #を使うときは文字列は""でかこんで記入します. "は CHR$(34)ですから, 文字列 a$ をかくときは次のようになります.

```
PRINT #1,CHR$(34)+a$+CHR$(34)
```

または

```
PRINT #1,CHR$(34);a$;CHR$(34)
```

③ 書き込まれたデータの読み出しは普通どおりです.

読み出しプログラムの例とその結果の例は次のとおりです.

【参考プログラム例】

```
OPEN "meibo1.dat" FOR INPUT AS #2          ファイルのオープン
FOR i = 1 TO 6                             ファイルのデータの読み出し
    INPUT #2, a$, x%, y%                   と表示
    PRINT a$; x%; y%
NEXT i
CLOSE #2                                   ファイルのクローズ
END
```

【結　果】

```
オオヤマタロウ 38   1570
ウエダ゛ヨシオ 41   1300
クリタサフ゛ロウ 25   820
トクカ゛ワヨシオ 52   2300
カワカミタロウ 21   420
アマダ゛ヨシミツ 37   1210
```

【プログラム例】

```
DIM x%(6, 2) ―――――――――――――――――――― 配列の宣言
FOR i = 0 TO 5 ―――――――――――――――――┐ 配列へデータを読み込む
    READ a$(i), x%(i, 1), x%(i, 2)      │
NEXT i ―――――――――――――――――――――――――┘
k$ = CHR$(34) ――――――――――――――――――― "を K$に代入
OPEN "b:meibo2.dat" FOR OUTPUT AS #1 ―――― ファイルのオープン
```

```
FOR i = 0 TO 5 ─────────────── PRINT#を使ってファイルへデータを
   PRINT #1, k$ + a$(i) + k$; x%(i, 1); x%(i, 2)   書き込む
NEXT i
CLOSE #1 ──────────────────── ファイルのクローズ
END
DATA ｵｵﾔﾏﾀﾛｳ,38,1570,ｳｴﾀﾞﾖｼｵ,41,1300
DATA ｸﾘﾀｻﾌﾞﾛｳ,25,820,ﾄｸｶﾞﾜﾖｼｵ,52,2300
DATA ｶﾜｶﾐﾀﾛｳ,21,420,ｱﾏﾀﾞﾖｼﾐﾂ,37,1210
```

【結　果】　データをファイルへ書き込みます.

〔演習問題〕

(152)　下のデータをシーケンシャルファイル yakyu.dat にかくプログラムをつくれ.

名　前	ホームラン	打　率
高田　四郎	15	.325
鈴木　太郎	25	.298
戸田　勝美	22	.268
徳田　義男	12	.352

(153)　(152) のデータを読み出して表示するプログラムをつくれ.

(154)　(152) のデータに追加して次のデータをかくプログラムをつくれ.

名　前	ホームラン	打　率
佐々木歳男	35	.275
大木　伸男	31	.333

(155)　(154) のデータを EOF を使って読み出して表示するプログラムをつくれ.

(156)　(154) の 2 番目のデータを高田広, 26, .299 で変更するプログラムをつくれ.

(157)　結果のデータ文のデータをシーケンシャルファイル"seisu.dat にかくプログラムをつくれ. ただし, 文字列データとしてかくものとする.

(158)　(157) のファイルを使って, 全体の和と平均を求めるプログラムをつくれ.

(159)　(157) のファイルを使って, 整数 x%を入力し, x%がいくつあるかを表示するプログラムをつくれ.

(160)　(157) のファイルを使って, 整数 xx%を入力し, xx%より大きいデータを表示するプログラムをつくれ.

〔解　答〕

(152)【プログラム例】

```
DIM a$(4), x%(4), y!(4) ──────────配列の宣言
FOR i = 0 TO 3 ─────────────────┐データの読み込み
   READ a$(i), x%(i), y!(i)      │
NEXT i ─────────────────────────┘
OPEN "b:yakyu.dat" FOR OUTPUT AS #1 ───ファイルのオープン
FOR i = 0 TO 3 ─────────────────┐データの書き込み
   WRITE #1, a$(i), x%(i), y!(i) │
NEXT i ─────────────────────────┘
CLOSE #1 ─────────────────────────ファイルのクローズ
END
DATA ﾀｶﾀﾞｼﾛｳ,15,.325,ｽｽﾞｷﾀﾛｳ,25,.298
DATA ﾄﾀﾞｶﾂﾐ,22,.268,ﾄｸﾀﾞﾖｼｵ,12,.352
```

【結　果】　データがファイルへ書き込まれます.

(153) に読み出し方と結果があります.

(153)【プログラム例】

```
OPEN "b:yakyu.dat" FOR INPUT AS #2 ───ファイルのオープン
FOR i = 1 TO 4 ─────────────────┐データの読み出しと表示
   INPUT #2, a$, x%, y!          │
   PRINT a$; x%; y!              │
NEXT i ─────────────────────────┘
CLOSE #2 ─────────────────────────ファイルのクローズ
END
```

【結　果】

```
ﾀｶﾀﾞｼﾛｳ 15  .325
ｽｽﾞｷﾀﾛｳ 25  .298
ﾄﾀﾞｶﾂﾐ 22  .268
ﾄｸﾀﾞﾖｼｵ 12  .352
```

(154)【プログラム例】

```
DIM a$(2), x%(2), y!(2) ──────────配列の宣言
FOR i = 0 TO 1 ─────────────────┐データの読み込み
   READ a$(i), x%(i), y!(i)      │
NEXT i ─────────────────────────┘
OPEN "b:yakyu.dat" FOR APPEND AS #3 ───ファイルのオープン
FOR i = 0 TO 1 ─────────────────┐データの追加書き込み
   WRITE #3, a$(i), x%(i), y!(i) │
NEXT i ─────────────────────────┘
CLOSE #3 ─────────────────────────ファイルのクローズ
END
DATA ｻｻｷﾄｼｵ,35,.275,ｵｵｷﾉﾌﾞｵ,31,.333
```

【結　果】　データが追加書き込みされます.

(155) に読み出し方と結果があります.

(155)【プログラム例】

```
OPEN "b:yakyu.dat" FOR INPUT AS #1 ——— ファイルのオープン
DO UNTIL EOF(1) ————————————————————— データの読み出しと表示
    INPUT #1, a$, x%, y!
    PRINT a$; x%; y!
LOOP
CLOSE #1 ———————————————————————————— ファイルのクローズ
END
```

【結　果】

```
ﾀｶﾀﾞｼﾛｳ 15  .325
ｽｽﾞｷﾀﾛｳ 25  .298
ﾄﾀﾞｶﾂﾐ 22  .268
ﾄｸﾀﾞﾖｼｵ 12  .352
ｻｻｷﾄｼｵ 35  .275
ｵｵｷﾉﾌﾞｵ 31  .333
```

(156)【プログラム例】

```
DIM aa$(100), xx%(100), yy!(100) ———— 配列の宣言
i = 0 ——————————————————————————————— データ組数を求めるカウンタ
OPEN "b:yakyu.dat" FOR INPUT AS #2 —— ファイルのオープン
DO UNTIL EOF(2) ————————————————————— データを読み出して配列へ代入
    i = i + 1
    INPUT #2, a$, x%, y!
    aa$(i) = a$
    xx%(i) = x%
    yy!(i) = y!
LOOP
CLOSE #2 ———————————————————————————— ファイルのクローズ
aa$(2) = "ﾀｶﾀﾞﾋﾛｼ" —————————————————— 新しいデータを代入
xx%(2) = 26
yy!(2) = .299
OPEN "b:yakyu.dat" FOR OUTPUT AS #1 — ファイルのオープン
FOR j = 1 TO i —————————————————————— データの書き込み
    WRITE #1, aa$(j), xx%(j), yy!(j)
NEXT j
CLOSE #1 ———————————————————————————— ファイルのクローズ
END
```

【結　果】（ファイルの内容）

```
ﾀｶﾀﾞｼﾛｳ 15  .325
ﾀｶﾀﾞﾋﾛｼ 26  .299
ﾄﾀﾞｶﾂﾐ 22  .268
ﾄｸﾀﾞﾖｼｵ 12  .352
ｻｻｷﾄｼｵ 35  .275
ｵｵｷﾉﾌﾞｵ 31  .333
```

(157)【プログラム例】

```
DIM x$(50) ────────────────────────データを入れる配列の宣言
FOR i = 0 TO 49 ──────────────────┬─データを配列 x$に代入
    READ x$(i)                    │
NEXT i ───────────────────────────┘
OPEN "b:seisu.dat" FOR OUTPUT AS #1───ファイルのオープン
FOR i = 0 TO 49 ──────────────────┬─データをファイルにかく
    WRITE #1, x$(i)               │
NEXT i ───────────────────────────┘
CLOSE #1 ──────────────────────────ファイルのクローズ
END
DATA 38,25,62,47,15,35,24,62,95,10
DATA 85,21,48,32,61,12,40,88,24,12
DATA 15,23,45,71,55,69,21,11,45,12
DATA 21,42,12,44,15,35,40,52,11,14
DATA 86,41,25,45,65,77,48,53,51,55
```

【結　果】

```
A>type seisu.dat     MS-DOS の TYPE 命令でファイル seisu.dat の内容を表示
"38"
"25"
"62"
"47"
"15"
"35"
"24"
"62"
"95"
"10"
以下省略
```

(158)【プログラム例】

```
OPEN "b:seisu.dat" FOR INPUT AS #1────ファイルのオープン
tt% = 0 ──────────────────────────────和を求める変数 tt%
FOR i = 0 TO 49 ─────────────────────┬─ファイルのデータを読み数値化して和を求める
    INPUT #1, x$                     │
    tt% = tt% + VAL(x$)              │
NEXT i ──────────────────────────────┘
CLOSE #1 ─────────────────────────────ファイルのクローズ
hi = tt% / 50──────────────────────────────平均を hi に求める
PRINT "ﾜ= "; tt%─────────────────────┬─────結果の表示
PRINT "ﾍｲｷﾝ= "; hi ──────────────────┘
END
```

【結　果】

```
ﾜ= 2035
ﾍｲｷﾝ= 40.7
```

(159)【プログラム例】

```
OPEN "b:seisu.dat" FOR INPUT AS #1 ──── ファイルのオープン
tt% = 0 ──── カウンタを 0 とする
INPUT x% ──── x%の入力
FOR i = 0 TO 49 ──── x%と同じ値のとき tt%に 1 を加える
    INPUT #1, x$
    IF x% = VAL(x$) THEN tt% = tt% + 1
NEXT i
CLOSE #1
PRINT x%; tt%; "コ" ──── 結果の表示
END
```

【結　果】

```
? 12                    12 を入力の例
 12   4 コ
? 11                    11 を入力の例
 11   2 コ
? 21                    21 を入力の例
 21   3 コ
```

(160)【プログラム例】

```
OPEN "b:seisu.dat" FOR INPUT AS #1 ──── ファイルのオープン
INPUT x% ──── x%の入力
FOR i = 0 TO 49 ──── x%をこえるデータを表示
    INPUT #1, x$
    IF x% < VAL(x$) THEN PRINT x$; " ";
NEXT i
CLOSE #1
PRINT
END
```

【結　果】

```
? 50                                                   50 を入力の例
62 62 95 85 61 88 71 55 69 52 86 65 77 53 51 55
? 80                                                   80 を入力の例
95 85 88 86
```

21章　ランダムファイル (1)

OPEN〜FOR RANDOM, FIELD, LSET, RSET, MKI$, MKS$, MKL$, MKD$, CVI, CVS, CVL, CVD, PUT#, GET#, CLOSE#

【例題 151】　ランダムファイル

【例題 145】のデータをランダムファイルにかくプログラムをつくれ.

【解　説】　◎　ランダムファイルにデータを書き込みます.

①　ランダムファイルは書き込み，読み出し，追加の区分はなくすべて FOR RANDOM でオープンします.

```
OPEN  "meibo3.dat" FOR RANDOM AS #1
```

- OPEN ← ファイルのオープン
- "meibo3.dat" ← ファイル名
- FOR RANDOM ← ランダムファイルの指定
- #1 ← オープン番号

②　ランダムファイルへの書き込みは Field 文を使ってフィールドを指定しバッファにデータをかいてから，ディスクへ書き込みます. 1 バッファは 256 バイトなので，複数組のデータを 1 バッファにつめてかくようにします. この方法はブロッキングによる書き込みといいます.

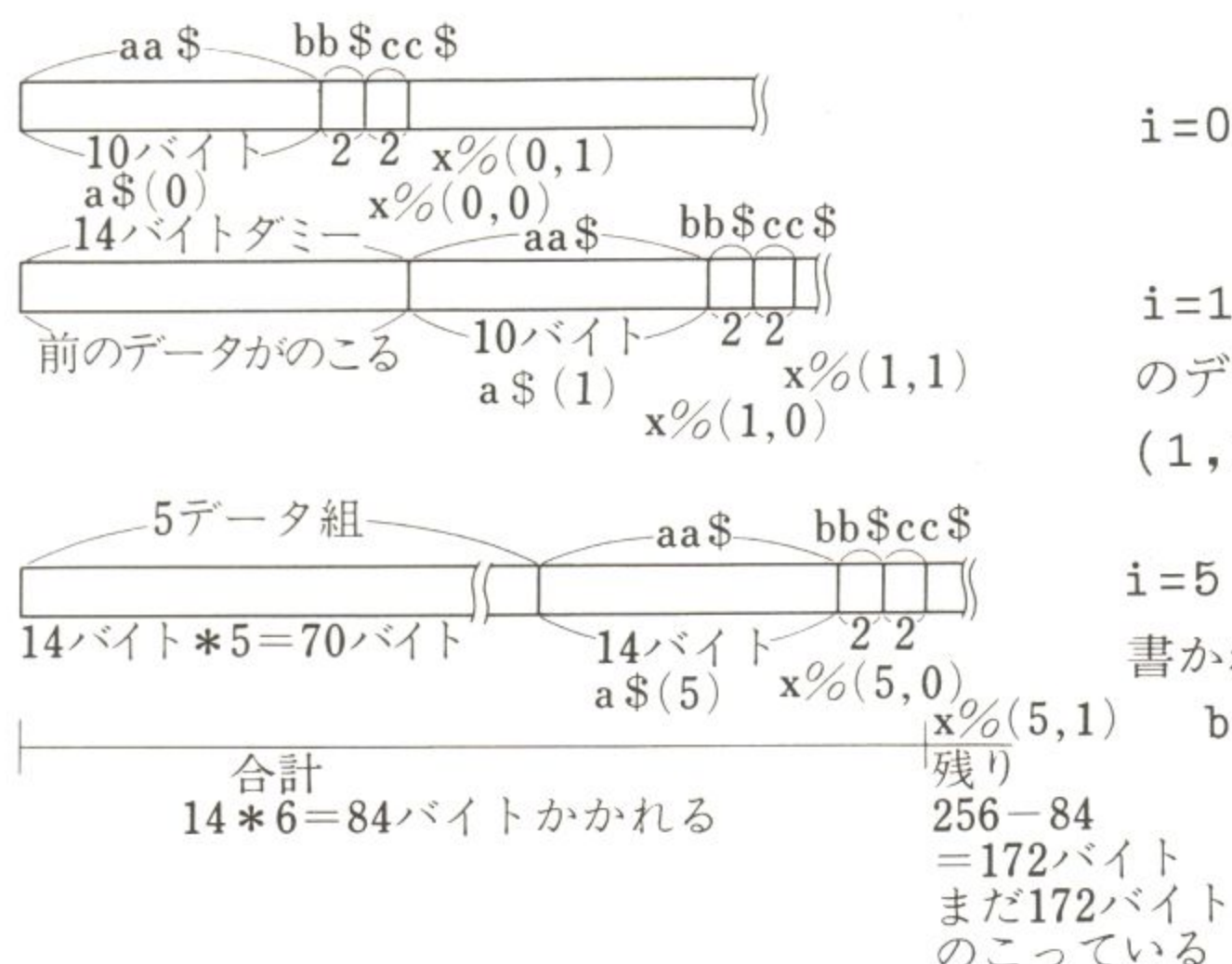
```
FOR i=0 TO  5
   FIELD  #1,14*i AS dd$,10 AS aa$,2 as bb$,2 as cc$
   LSET aa$=a$(i)
   LSET bb$=MKI$(x%(i,0))
   LSET cc$=MKI$(x%(i, 1))
NEXT i
PUT#1,1
```

- 10 AS aa$,2 as bb$,2 as cc$ ← フィールドの分割. ③で説明
- LSET aa$=a$(i) ← a$(i)を aa$に代入
- LSET bb$=…, LSET cc$=… ← x%(i,0)， x%(i,1)を bb$,cc$に代入
- MKI$ ← 整数を文字化する. 長整数は MKL$, 単精度実数は MKS$, 倍精度実数は MKD$を使う.
- PUT#1,1 ← 第 1 レコードとしてディスクへ書き込み
- LSET ← 左づめにセット，右づめの時は RSET

③　フィールドの分割は次のようになります.

aa $　bb $ cc $
10バイト　2 2　x%(0,1)
a $(0)　x%(0,0)

14バイトダミー　aa $　bb $ cc $
前のデータがのこる　10バイト　2 2　x%(1,1)
a $ (1)　x%(1,0)

5データ組　aa $　bb $ cc $
14バイト＊5＝70バイト　14バイト　2 2
a $(5)　x%(5,0)　x%(5,1)
合計
14＊6＝84バイトかかれる
残り
256－84
＝172バイト
まだ172バイト
のこっている

i=0 のとき aa$は 0 のため aa$, bb$, cc$を書く

i=1 のとき aa$は 14 のため最初の 14 バイトは何も書かれず前のデータがのこる．次の 14 バイトに a$(1), x%(1,0), x%(1,1)が書き込まれる．

i=5 のとき aa$は 70 バイトとなり，最初の 70 バイトには何も書かれず前のデータがのこる．71 バイト目から aa$(a$(5)), bb$(x%(5,0)), cc$(x%(5,1))が書かれる．

④ ブロッキングを使わないと，FIELD #1, 10 AS aa$, 2 AS bb$, 2 AS cc$となり，1組のデータを aa$, bb$, cc$にセットして PUT #1, 1 から PUT #1, 6 まで 6 レコードに分けて格納します．256 バイトの中にわずか 14 バイトしか使わない無駄となります．

```
    ≀
FOR i=0  TO  5
        FIELD  10 AS aa$,2 AS bb$,2 AS cc$
        LSET aa$=a$(i)
        LSET bb$=MKI$(x%(i,0))
        LSET cc$=MKI$(x%(i,1))
        PUT  #1,i+1
NEXT i
    ≀
```

⑤ なお，2 AS bb$, bb$=MKI$(x%(i,0))のように整数は MKI$を使って文字変数に代入します．その場合の必要バイト数は 2 です．単精度実数の場合は 4 AS bb$, bb$=MKS$(x!(i,0))のように MKS$を使って文字変数に代入します．必要バイト数は 4 です．

長整数の場合は MKL$を使います．必要バイト数は 4 です．

倍精度実数の場合は MKD$を使います．必要バイト数は 8 です．

【プログラム例】

```
DIM a$(18), X%(18, 2) ──────────── A$は名前，x%は年齢と年収の入る配列
FOR i = 0 TO 17 ─────────────┐── データの配列への読み込み
    IF i <= 5 THEN             │
        READ a$(i), X%(i, 0), X%(i, 1)
    ELSE                       │   i が 6 以上は a$"",
        a$(i) = ""             │   x%は 0 となる
        X%(i, 0) = 0           │
        X%(i, 1) = 0           │
    END IF                     │
NEXT i ────────────────────────┘
OPEN "b:meibo3.dat" FOR RANDOM AS #1 LEN = 256 ──────── ファイルのオープン
```

```
FOR i = 0 TO 17 ──────────────────────── ファイルへ書き込み
    FIELD #1, 14 * i AS dd$, 10 AS aa$, 2 AS bb$, 2 AS cc$
    LSET aa$ = a$(i)
    LSET bb$ = MKI$(X%(i, 0))
    LSET cc$ = MKI$(X%(i, 1))
NEXT i
PUT #1, 1
CLOSE #1 ─────────────────────────────── ファイルのクローズ
END
DATA ｵｵﾔﾏﾀﾛｳ,38,1570,ｳｴﾀﾞﾖｼｵ,41,1300
DATA ｸﾘﾀｻﾌﾞﾛｳ,25,820,ﾄｸｶﾞﾜﾖｼｵ,52,2300
DATA ｶﾜｶﾐﾀﾛｳ,21,420,ｱﾏﾀﾞﾖｼﾐﾂ,37,1210
```

【例題 152】　ランダムファイルの読み出し

【例題 151】のファイルのデータを読み出し表示するプログラムをつくれ．

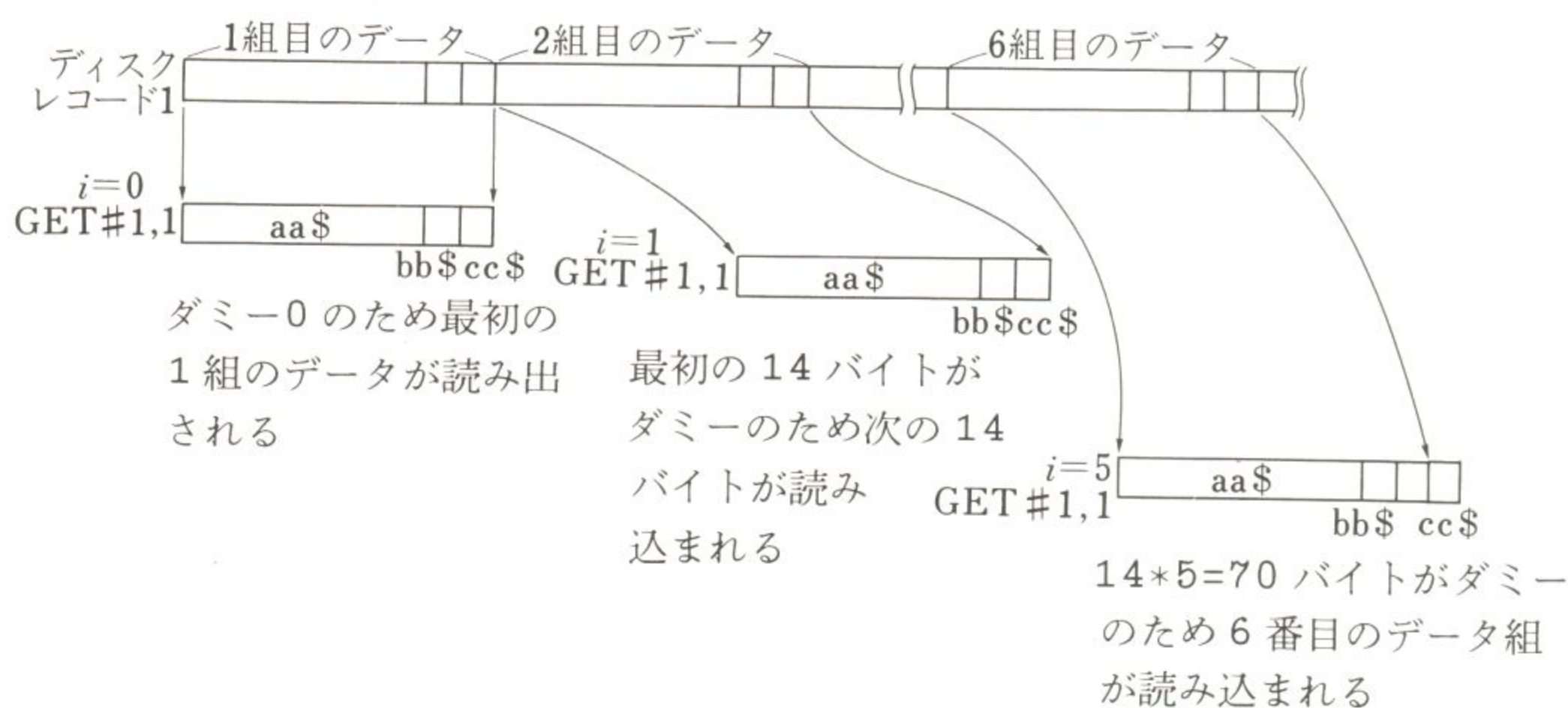
【解　説】　◎　ランダムファイルのデータを読み出します．

① ランダムファイルの読み出しはまず書き込みと同じ方法でファイルをオープンします．

② 次に書き込みと同じ方法で FIELD を割りあてます．

③ GET　#を使ってディスクからデータを読み出します．

GET　#1, 1 によってレコード 1 を読み出します．FIELD が【例題 151】と同じとすると i=0 のとき最初の 1 組のデータが aa$, bb$, cc$に読み出されます．i=1 のときは最初の 14 バイトがダミーで次の 14 バイトすなわち 2 組目のデータが aa$, bb$, cc$に読み出されます．

④ 整数化は CVI を使います．単精度実数の場合は CVS を使います．長整数の場合は CVL，倍精度実数の場合は CVD を使います．

【プログラム例】

```
DIM a$(18), x%(18, 2) ──────────────── 配列 a$は名前，x%は年齢と年収が入る
OPEN "b:meibo3.dat" FOR RANDOM AS #1 LEN = 256 ──── ファイルのオープン
```

```
FOR i = 0 TO 5 ──────────────────────── 読み出し
    FIELD #1, 14 * i AS dd$, 10 AS aa$, 2 AS bb$, 2 AS cc$
    GET #1, 1
    a$(i) = aa$
    x%(i, 0) = CVI(bb$)
    x%(i, 1) = CVI(cc$)
NEXT i
CLOSE #1 ─────────────────────────────── ファイルのクローズ
FOR i = 0 TO 5 ──────────────────────── 表示
    PRINT a$(i); x%(i, 0); x%(i, 1)
NEXT i
END
```

【結　果】

```
ｵｵﾔﾏﾀﾛｳ      38   1570
ｳｴﾀﾞﾖｼｵ      41   1300
ｸﾘﾀｻﾌﾞﾛｳ     25   820
ﾄｸｶﾞﾜﾖｼｵ     52   2300
ｶﾜｶﾐﾀﾛｳ      21   420
ｱﾏﾀﾞﾖｼﾐﾂ     37   1210
```

【例題 153】　ランダムファイルへデータの追加

【例題 151】のファイルに下のデータを追加するプログラムをつくれ.

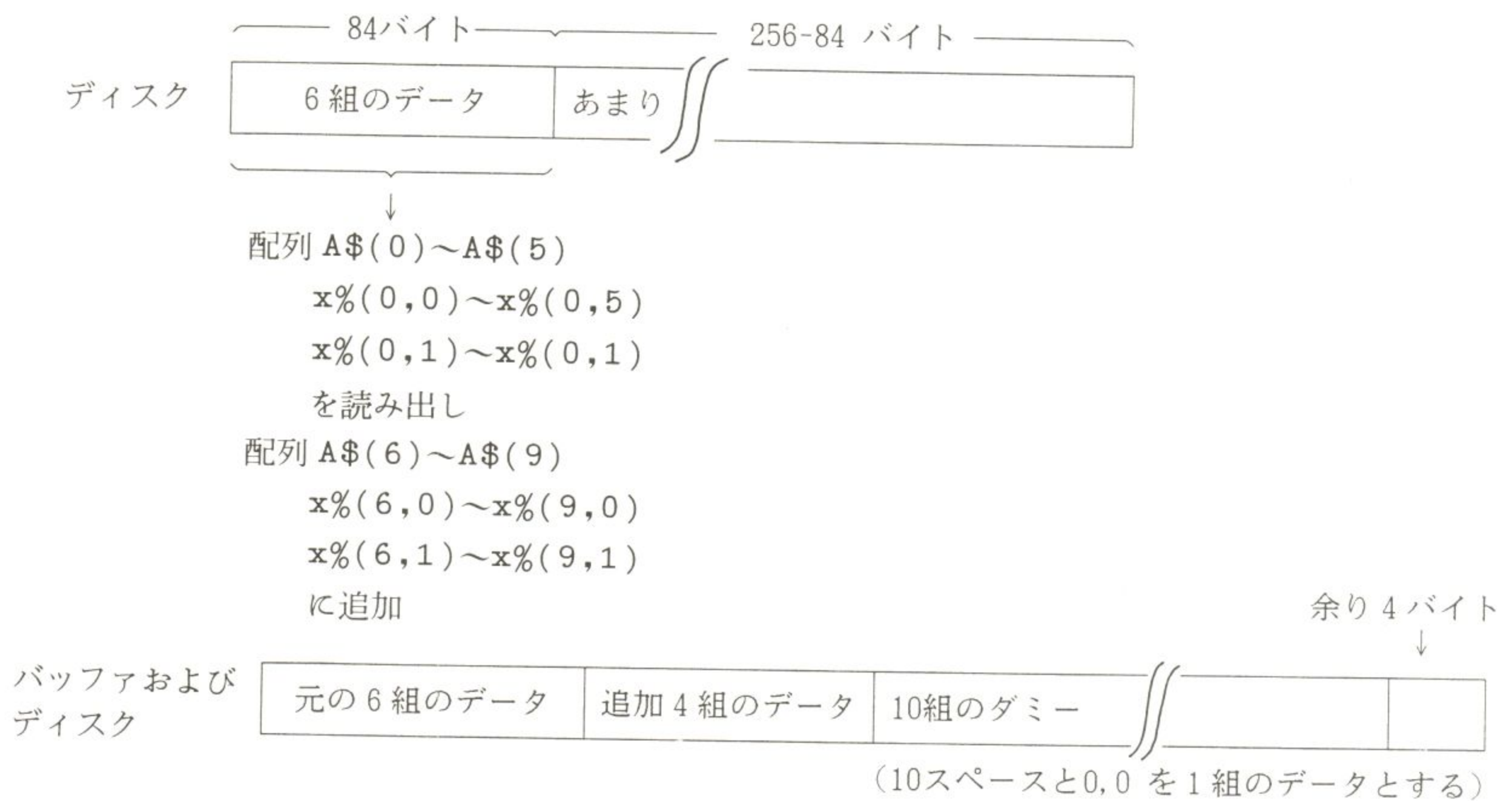
名　前	年　齢	年　収
川口　　等	26	550
太田　浩一	51	1820
岸田　公吉	33	710
和田　吉男	38	830

【解　説】　◎　ランダムファイルへデータを追加

① 【例題 151】のランダムファイルは 1 組のデータが文字列 10 文字, 整数型, 整数型で合計 14 バイトを使用します. したがって, 1 バッファ256¥14=18 により 18 データ組が 1 バッファに入ります.

② 【例題 151】では 6 データ組のみ入っています. したがって, 今回の 4 データ組をその後に追加します.【例題 151】の 6 組と今回の 4 組のデータをかき, 後の 8 組はスペースの文字列と数値 0 を記入しておきます.

③ 1 度ファイル meibo 3.dat のデータを読み出し, 配列に格納します. その後に 4 データ組をつけ加えます.

④ 合計 10 組をバッファにかきなおします. 後の 8 組はスペースと 0 をかき, バッファ内容 18 組のデータをディスクへ格納します.

【プログラム例】

```
DIM a$(18), x%(18, 2)                                    ─配列 a$は名前, x%は年齢と年収が入る
OPEN "b:meibo3.dat" FOR RANDOM AS #1 LEN = 256           ─ファイルのオープン
FOR i = 0 TO 17                                          ─0〜17 組のデータの配列への読み
    FIELD #1, 14 * i AS dd$, 10 AS aa$, 2 AS bb$, 2 AS cc$   出し
    GET #1, 1
    a$(i) = aa$
    IF a$(i) = "          " THEN EXIT FOR
    x%(i, 0) = CVI(bb$)
    x%(i, 1) = CVI(cc$)
NEXT i
j = i                                                    ─データ組の追加
FOR i = j TO j + 3
   READ a$(i), x%(i, 0), x%(i, 1)
NEXT i
FOR i = 0 TO j + 3                                       ─追加データ組をディスケットへ
    FIELD #1, 14 * i AS dd$, 10 AS aa$, 2 AS bb$, 2 AS cc$   書き込む
    LSET aa$ = a$(i)
    LSET bb$ = MKI$(x%(i, 0))
    LSET cc$ = MKI$(x%(i, 1))
NEXT i
PUT #1, 1                                                ─ファイルのクローズ
CLOSE #1
END
DATA ｶﾜｸﾞﾁﾋﾄｼ,26,550,ｵｵﾀｺｳｲﾁ,51,1820
DATA ｷｼﾀﾞｺｳｷﾁ,33,710,ﾜﾀﾞﾖｼｵ,38,830
```

【例題 154】　ランダムファイルの読み出し

【例題 153】のファイル meibo3.dat を読み出して表示するプログラムをつくれ.

【解　説】　◎　ランダムファイルの読み出し

① 0〜18 組のデータを読み出し配列へ代入します. その中でデータの入っていないところは名前は

10 文字の空白，年齢と年収は 0 ですから，それをみつければどこまでデータがあるかわかります.

```
FOR i=0  TO  17
        FIELD  14*i AS aa$,  10 AS aa$,  2 AS bb$,  2 AS cc$
        GET  #1, 1                    —— 第１レコードをよみ出す.
        a$(i)=aa$                     —— aa$に名前が読み出されそれを a$(i)に代入
        IF a$(i)="        " THEN EXIT FOR —— a$(i)が空白のときループの外へ
        x%(i,0)=CVI(bb$)              ┐ 名前が空白でないとき年齢を x%（i, 0),
        x%(i,1)=CVI (cc$)             ┘ 年収を x%(i,1)へ代入
NEXT i
                                      データの数は i 個
```

【プログラム例】

```
DIM a$(18), x%(18, 2)—————————————— 名前の入る a$, 年齢, 年収が x%に入る
OPEN "b:meibo3.dat" FOR RANDOM AS #1 LEN = 256——ファイルのオープン
FOR i = 0 TO 17 ————————————————————┐ データ組がなくなるまで名前を a$(i),
   FIELD #1, 14 * i AS dd$, 10 AS aa$, 2 AS bb$, 2 AS cc$  年齢を x%(i,0), 年収を x%(i,1)に
   GET #1, 1                                              代入
   a$(i) = aa$
   IF a$(i) = "          " THEN EXIT FOR
   x%(i, 0) = CVI(bb$)
   x%(i, 1) = CVI(cc$)
NEXT i ——————————————————————————————┘
j = i
CLOSE #1
FOR i = 0 TO j - 1——————————————————┐ データ組の表示
  PRINT a$(i); x%(i, 0); x%(i, 1)
NEXT i——————————————————————————————┘
END
```

【結　果】

```
オオヤマタロウ      38   1570
ウエダ゛ヨシオ      41   1300
クリタサブ゛ロウ    25   820
トクガ゛ワヨシオ    52   2300
カワカミタロウ      21   420
アマダ゛ヨシミツ    37   1210
カワグ゛チヒトシ    26   550
オオタコウイチ      51   1820
キシダ゛コウキチ    33   710
ワダ゛ヨシオ        38   830
```

【例題 155】　ランダムファイルの訂正

【例題 153】のファイルの 2 番目を下のデータで書きかえてから全体を表示するプログラムをつくれ.

名前：羽田　修

年齢： 26 歳

年収： 690 万円

【解　説】　◎　ランダムファイルのデータの変更

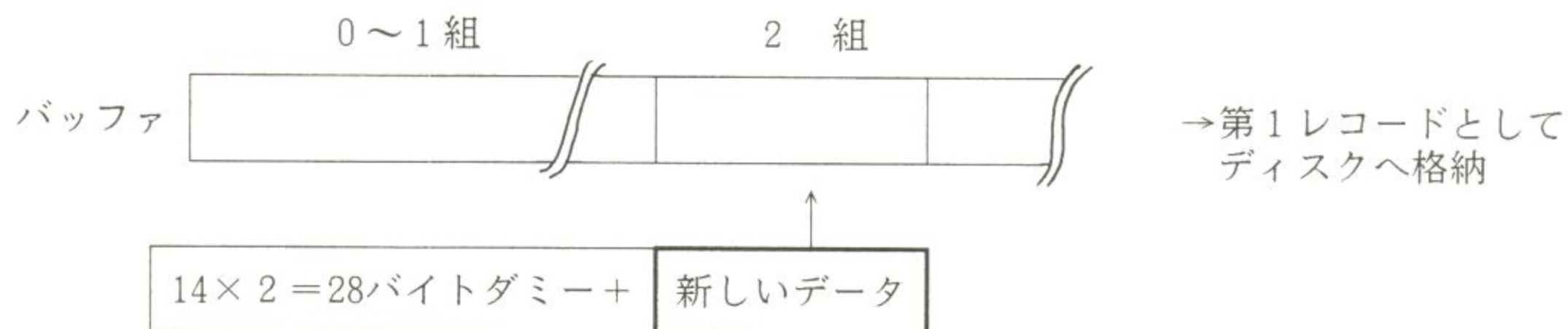
① ランダムファイルのデータの変更は比較的簡単です．

② まず 2 番目のデータ組が何番目のレコードにあるかをさがします．次にそのレコードの何組目かをみつけます．そして，そのレコード番号のみつかった順番のデータ組を前にダミーをつけてバッファにかきこみ，そのレコードをディスクへ格納します．

本例では 1 レコードに 18 データ組が入っていますから 0〜17 組が第 1 レコード，18〜35 組が第 2 レコード…となります．2 組目は第 1 レコードの 2 番目となります．

0〜1 組　　2　組

バッファ　→第 1 レコードとして ディスクへ格納

14× 2 ＝28バイトダミー＋　新しいデータ

③ 一般化します．1 レコードは 18 データ組（0〜17 組）が入っていて，第 1 レコードから順にレコード番号があるとします．データ組は 0 番目から始まり，レコードは 1 番目から始まるとします．この時，x 番目のデータ組は y+1 レコードの z 番目となります．

x÷18=y あまり z

例　23 番目は　23÷18=1 あまり 5　　したがって 2 レコードの 5 番目

145 番目は 145÷18=8 あまり 1　　したがって 9 レコードの 1 番目

【プログラム例】

```
DIM a$(18), x%(18, 2) ──────── 0〜17 組のデータ組の入る配列
r = 2 ¥ 18 + 1 ──────── 2 番目のレコード組が所属するレコード数
b = 2 MOD 18 ──────── 2 番目のレコード組が r レコードの b 番目に属する
OPEN "b:meibo3.dat" FOR RANDOM AS #1 LEN = 256   ファイルのオープン
FOR i = 0 TO 17 ──────┐ r レコードのデータ組を読み出し
   FIELD #1, 14 * i AS dd$, 10 AS aa$, 2 AS bb$, 2 AS cc$ │ 配列に代入
   GET #1, r
   a$(i) = aa$
   x%(i, 0) = CVI(bb$)
   x%(i, 1) = CVI(cc$)
NEXT i ──────┘
a$(b) = "ﾊﾈﾀﾞ ｵｻﾑ" ──────┐ 訂正データを配列に代入
x%(b, 0) = 26
x%(b, 1) = 690 ──────┘
FOR i = 0 TO 17 ──────┐ r レコードに訂正データを含む
   FIELD #1, 14 * i AS dd$, 10 AS aa$, 2 AS bb$, 2 AS cc$ │ 0〜17 組のデータを書き込む
   LSET aa$ = a$(i)
   LSET bb$ = MKI$(x%(i, 0))
   LSET cc$ = MKI$(x%(i, 1))
NEXT i
PUT #1, r ──────┘
CLOSE #1 ──────── ファイルのクローズ
END
```

【結　果】

ｵｵﾔﾏﾀﾛｳ	38	1570	3 番目を変更
ｳｴﾀﾞﾖｼｵ	41	1300	
ﾊﾈﾀﾞｵｻﾑ	26	690	
ﾄｸｶﾞﾜﾖｼｵ	52	2300	
ｶﾜｶﾐﾀﾛｳ	21	420	
ｱﾏﾀﾞﾖｼﾐﾂ	37	1210	
ｶﾜｸﾞﾁﾋﾄｼ	26	550	
ｵｵﾀｺｳｲﾁ	51	1820	
ｷｼﾀﾞｺｳｷﾁ	33	710	
ﾜﾀﾞﾖｼｵ	38	830	

【例題 156】　データの削除

【例題 155】のデータを使い，太田浩一のデータを削除するプログラムをつくれ．

【解　説】　◎　データの削除をします．

①　データの削除は，データを全くなくして，空になった部分をつめてしまう方法や，削除するデータに削除を示すマークをつける方法などがあります．ここでは比較的簡単な削除マークをつける方法を行います．これは全く削除してしまう方法に比べ，必要によって復活できるメリットもあります．

②　データを読み出し，削除するデータの名前の先頭に"*"マークをつけます．これによって，このデータ組は削除されることがわかります．

③　"*"マークをつけたデータを含めてファイルをかきかえます．

【プログラム例】

```
DIM a$(18), x%(18, 2)          データを読み出してオオタコウイチのとき*を先頭につける
OPEN "b:meibo3.dat" FOR RANDOM AS #1 LEN = 256
FOR i = 0 TO 17
    FIELD #1, 14 * i AS dd$, 10 AS aa$, 2 AS bb$, 2 AS cc$
    GET #1, 1
    a$(i) = aa$
    IF a$(i) = "          " THEN EXIT FOR
    IF a$(i) = "ｵｵﾀｺｳｲﾁ   " THEN a$(i) = "*ｵｵﾀｺｳｲﾁ  "
    x%(i, 0) = CVI(bb$)
    x%(i, 1) = CVI(cc$)
NEXT i
j = i
FOR i = 0 TO i - 1                                      ファイルのかきなおし
    FIELD #1, 14 * i AS dd$, 10 AS aa$, 2 AS bb$, 2 AS cc$
    LSET aa$ = a$(i)
    LSET bb$ = MKI$(x%(i, 0))
    LSET cc$ = MKI$(x%(i, 1))
NEXT i
PUT #1, 1
CLOSE #1
END
```

【結　果】（ファイルの内容）

ｵｵﾔﾏﾀﾛｳ	38	1570
ｳｴﾀﾞﾖｼｵ	41	1300
ﾊﾈﾀﾞｵｻﾑ	26	690

```
トクガ゛ワヨシオ    52   2300
カワカミタロウ      21   420
アマダ゛ヨシミツ    37   1210      全体を表示するとオオタコウイチは先頭に*がついている
カワグ゛チヒトシ    26   550
*オオタコウイチ     51   1820
キシダ゛コウキチ    33   710
ワダ゛ヨシオ        38   830
```

> **【例題 157】 削除したデータを除いてリスト**
>
> 【例題 156】のデータの一覧表を表示するプログラムをつくれ.

【解　説】 ◎ 削除データを除いて表示します.

① 削除データは名前の先頭に"*"がついていますので，名前の先頭をみて"*"であれば表示しません.

【プログラム例】

```
DIM a$(18), x%(18, 2)
OPEN "b:meibo3.dat" FOR RANDOM AS #1 LEN = 256
FOR i = 0 TO 17 ──────────────────────────────────────┐─データの読み出し
    FIELD #1, 14 * i AS dd$, 10 AS aa$, 2 AS bb$, 2 AS cc$
    GET #1, 1
    a$(i) = aa$
    IF a$(i) = "          " THEN EXIT FOR
    x%(i, 0) = CVI(bb$)
    x%(i, 1) = CVI(cc$)
NEXT i ───────────────────────────────────────────────┘
j = i
CLOSE #1
FOR i = 0 TO j - 1 ───────────────────────────────────┐─表示，名前の先頭が
   IF LEFT$(a$(i), 1) <> "*" THEN PRINT a$(i); x%(i, 0); x%(i, 1)   "*"のとき非表示
NEXT i ───────────────────────────────────────────────┘
END
```

【結　果】

```
オオヤマタロウ      38   1570
ウエダ゛ヨシオ      41   1300
ハネダ゛オサム      26   690
トクガ゛ワヨシオ    52   2300
カワカミタロウ      21   420
アマダ゛ヨシミツ    37   1210
カワグ゛チヒトシ    26   550
キシダ゛コウキチ    33   710
ワダ゛ヨシオ        38   830
```

> **【例題 158】 削除データの表示**
>
> 【例題 156】で削除したデータのリストを表示するプログラムをつくれ.

【解　説】 ◎ 削除データを表示します.

① 【例題 157】の逆で，名前の先頭に"*"がついているものを表示します.

【プログラム例】

```
DIM a$(18), x%(18, 2)
OPEN "b:meibo3.dat" FOR RANDOM AS #1 LEN = 256
FOR i = 0 TO 17 ─────────────────────────── データのよみ出し
    FIELD #1, 14 * i AS dd$, 10 AS aa$, 2 AS bb$, 2 AS cc$
    GET #1, 1
    a$(i) = aa$
    IF a$(i) = "          " THEN EXIT FOR
    x%(i, 0) = CVI(bb$)
    x%(i, 1) = CVI(cc$)
NEXT i ─────────────────────────────────────
j = i
CLOSE #1
FOR i = 0 TO j - 1 ──────────────────────── *のついているもの
   IF LEFT$(a$(i), 1) = "*" THEN PRINT a$(i); x%(i, 0); x%(i, 1)   のみ表示
NEXT i ─────────────────────────────────────
END
```

【結　果】

```
*オオタコウイチ    51   1820
```

〔演習問題〕

(161)　次のデータをランダムファイル meibo4.dat にかくプログラムをつくれ．ただし，名前は 16 文字，出身地と所属は 10 文字分とるものとし，1 レコードに 4 組書き込むのとする．

名　前	年齢	出身地	所　属	年収（万円）
遠藤　一郎	33	北　海　道	営業部	465
池田　宏一	36	鹿 児 島 県	総務部	430
中田　浩二	29	大　阪　府	業務部	428
海田　泰二	43	山　口　県	貿易部	580
近藤　三郎	25	青　森　県	業務部	398
小田　憲男	40	富　山　県	経理部	411
山川　一夫	38	東　京　都	営業部	490
河田　安次	30	千　葉　県	営業部	450

(162)　(161) のデータを読み出し表示するプログラムをつくれ．

(163)　(161) のデータを読み出し，40 歳代の人を表示するプログラムをつくれ．

(164)　(161) のデータに次のデータを追加してかくプログラムをつくれ．

名　前	年齢	出身地	所　属	年収（万円）
江田　五郎	53	東　京　都	営業部	1250
小山　吾郎	46	愛　知　県	企画部	850
井沢　三郎	50	高　知　県	営業部	1000
高田　　広	53	福　岡　県	業務部	1150
大崎　　猛	45	北　海　道	営業部	920

(165)　(164) のデータを表示するプログラムをつくれ.

(166)　(164) のデータのうち高田広の出身地を福岡県から鹿児島県に変更するプログラムをつくれ.

(167)　(164) のデータのうち池田宏一のデータを削除するプログラムをつくれ.

〔解　答〕

(161)【プログラム例】

```
DIM a$(16, 3), X%(16, 2) ──────────── データの読み込み
FOR i = 0 TO 7
   READ a$(i, 0), X%(i, 0), a$(i, 1), a$(i, 2), X%(i, 1)
NEXT i
OPEN "b:meibo4.dat" FOR RANDOM AS #1 LEN = 180 ──── データのファイルへの書き込み
FOR i = 1 TO 2
   FOR j = 0 TO 3
      FIELD #1, 40 * j AS ddd$, 16 AS aa$, 2 AS bb$, 10 AS cc$, 10 AS dd$, 2 AS ee$
      k = (i - 1) * 4 + j
      LSET aa$ = a$(k, 0)
      LSET bb$ = MKI$(X%(k, 0))
      LSET cc$ = a$(k, 1)
      LSET dd$ = a$(k, 2)
      LSET ee$ = MKI$(X%(k, 1))
   NEXT j
PUT #1, i
NEXT i
CLOSE #1
END
DATA ｴﾝﾄﾞｳｲﾁﾛｳ,33,ﾎｯｶｲﾄﾞｰ,ｴｲｷﾞｮｳﾌﾞ,465
DATA ｲｹﾀﾞｺｳｲﾁ,36,ｶｺﾞｼﾏｹﾝ,ｿｳﾑﾌﾞ,430
DATA ﾅｶﾀﾞｺｳｼﾞ,29,ｵｵｻｶﾌ,ｷﾞｮｳﾑﾌﾞ,428
DATA ｶｲﾀﾞﾀｲｼﾞ,43,ﾔﾏｸﾞﾁｹﾝ,ﾎﾞｳｴｷﾌﾞ,580
DATA ｺﾝﾄﾞｳｻﾌﾞﾛｳ,25,ｱｵﾓﾘｹﾝ,ｷﾞｮｳﾑﾌﾞ,398
DATA ｵﾀﾞﾉﾘｵ,40,ﾄﾔﾏｹﾝ,ｹｲﾘﾌﾞ,411
DATA ﾔﾏｶﾜｶｽﾞｵ,38,ﾄｳｷｮｳﾄ,ｴｲｷﾞｮｳﾌﾞ,490
DATA ｶﾜﾀﾞﾔｽｼﾞ,30,ﾁﾊﾞｹﾝ,ｴｲｷﾞｮｳﾌﾞ,450
```

(162)【プログラム例】

```
DIM a$(16, 3), x%(16, 2)
OPEN "b:meibo4.dat" FOR RANDOM AS #1 LEN = 180 ─┬ ファイルからデータの読み出し
FOR i = 1 TO 2
    FOR j = 0 TO 3
        FIELD #1, 40 * j AS ddd$, 16 AS aa$, 2 AS bb$, 10 AS cc$, 10 AS dd$, 2 AS ee$
        GET #1, i
        k = (i - 1) * 4 + j
        a$(k, 0) = aa$
        x%(k, 0) = CVI(bb$)
        a$(k, 1) = cc$
        a$(k, 2) = dd$
        x%(k, 1) = CVI(ee$)
    NEXT j
NEXT i
CLOSE #1
FOR i = 0 TO 7 ─┬ 表示
    PRINT a$(i, 0); x%(i, 0), a$(i, 1); a$(i, 2); x%(i, 1)
NEXT i
END
```

【結　果】

```
ｴﾝﾄﾞｳｲﾁﾛｳ        33      ﾎｯｶｲﾄﾞｰ    ｴｲｷﾞｮｳﾌﾞ   465
ｲｹﾀﾞｺｳｲﾁ         36      ｶｺﾞｼﾏｹﾝ    ｿｳﾑﾌﾞ      430
ﾅｶﾀﾞｺｳｼﾞ         29      ｵｳｻｶﾌ      ｷﾞｮｳﾑﾌﾞ    428
ｶｲﾀﾞﾀｲｼﾞ         43      ﾔﾏｸﾞﾁｹﾝ    ﾎﾞｳｴｷﾌﾞ    580
ｺﾝﾄﾞｳｻﾌﾞﾛｳ       25      ｱｵﾓﾘｹﾝ     ｷﾞｮｳﾑﾌﾞ    398
ｵﾀﾞﾉﾘｵ           40      ﾄﾔﾏｹﾝ      ｹｲﾘﾌﾞ      411
ﾔﾏｶﾜｶｽﾞｵ         38      ﾄｳｷｮｳﾄ     ｴｲｷﾞｮｳﾌﾞ   490
ｶﾜﾀﾞﾔｽｼﾞ         30      ﾁﾊﾞｹﾝ      ｴｲｷﾞｮｳﾌﾞ   450
```

(163)【プログラム例】

```
DIM a$(16, 3), x%(16, 2)
OPEN "b:meibo4.dat" FOR RANDOM AS #1 LEN = 180 ─┬ データのファイルからの読み出し
FOR i = 1 TO 2
    FOR j = 0 TO 3
        FIELD #1, 40 * j AS ddd$, 16 AS aa$, 2 AS bb$, 10 AS cc$, 10 AS dd$, 2 AS ee$
        GET #1, i
        k = (i - 1) * 4 + j
        a$(k, 0) = aa$
        x%(k, 0) = CVI(bb$)
        a$(k, 1) = cc$
        a$(k, 2) = dd$
        x%(k, 1) = CVI(ee$)
    NEXT j
NEXT i
CLOSE #1
FOR i = 0 TO 7 ─┬ 40 歳代の人の表示
    IF x%(i, 0) < 50 AND x%(i, 0) >= 40 THEN
        PRINT a$(i, 0); x%(i, 0), a$(i, 1); a$(i, 2); x%(i, 1)
    END IF
NEXT i
END
```

【結 果】

カイタ゛タイシ゛	43	ヤマグ゛チケン	ボ゛ウエキブ゛	580
オタ゛ノリオ	40	トヤマケン	ケイリブ゛	411

(164)【プログラム例】

```
DIM a$(16, 3), X%(16, 2) ──────────────── データの配列への読み出し
FOR i = 0 TO 4
    READ a$(i, 0), X%(i, 0), a$(i, 1), a$(i, 2), X%(i, 1)
NEXT i
OPEN "b:meibo4.dat" FOR RANDOM AS #1 LEN = 180 ──── データのファイルへの書き込み
FOR i = 3 TO 4
    FOR j = 0 TO 3
        FIELD #1, 40 * j AS ddd$, 16 AS aa$, 2 AS bb$, 10 AS cc$, 10 AS dd$, 2 AS ee$
        k = (i - 3) * 4 + j
        LSET aa$ = a$(k, 0)
        LSET bb$ = MKI$(X%(k, 0))
        LSET cc$ = a$(k, 1)
        LSET dd$ = a$(k, 2)
        LSET ee$ = MKI$(X%(k, 1))
    NEXT j
PUT #1, i
NEXT i
CLOSE #1
END
DATA ｴﾀﾞｺﾞﾛｳ,53,ﾄｳｷｮｳﾄ,ｴｲｷﾞｮｳﾌﾞ,1250
DATA ｺﾔﾏｺﾞﾛｳ,46,ｱｲﾁｹﾝ,ｷｶｸﾌﾞ,850
DATA ｲｻﾞﾜｻﾌﾞﾛｳ,50,ｺｳﾁｹﾝ,ｴｲｷﾞｮｳﾌﾞ,1000
DATA ﾀｶﾀﾞﾋﾛｼ,53,ﾌｸｵｶｹﾝ,ｴｲｷﾞｮｳﾌﾞ,1150
DATA ｵｵｻｷﾂﾖｼ,45,ﾎｯｶｲﾄﾞｰ,ｴｲｷﾞｮｳﾌﾞ,920
```

(165)【プログラム例】

```
DIM a$(16, 3), x%(16, 2) ──────────────── ファイルのオープン
OPEN "b:meibo4.dat" FOR RANDOM AS #1 LEN = 180 ── データの読み出し
FOR i = 1 TO 4
    FOR j = 0 TO 3
        FIELD #1, 40 * j AS ddd$, 16 AS aa$, 2 AS bb$, 10 AS cc$, 10 AS dd$, 2 AS ee$
        GET #1, i
        k = (i - 1) * 4 + j
        a$(k, 0) = aa$
        x%(k, 0) = CVI(bb$)
        a$(k, 1) = cc$
        a$(k, 2) = dd$
        x%(k, 1) = CVI(ee$)
    NEXT j
NEXT i
CLOSE #1 ──────────────────────────── ファイルのクローズ
FOR i = 0 TO 15
    PRINT a$(i, 0); x%(i, 0), a$(i, 1); a$(i, 2); x%(i, 1) ── 表示
NEXT i
END
```

【結　果】

```
ｴﾝﾄﾞｳｲﾁﾛｳ        33   ﾎｯｶｲﾄﾞｰ    ｴｲｷﾞｮｳﾌﾞ   465
ｲｹﾀﾞｺｳｲﾁ         36   ｶｺﾞｼﾏｹﾝ    ｿｳﾑﾌﾞ      430
ﾅｶﾀﾞｺｳｼﾞ         29   ｵｳｻｶﾌ      ｷﾞｮｳﾑﾌﾞ    428
ｶｲﾀﾞﾀｲｼﾞ         43   ﾔﾏｸﾞﾁｹﾝ    ﾎﾞｳｴｷﾌﾞ    580
ｺﾝﾄﾞｳｻﾌﾞﾛｳ       25   ｱｵﾓﾘｹﾝ     ｷﾞｮｳﾑﾌﾞ    398
ｵﾀﾞﾉﾘｵ           40   ﾄﾔﾏｹﾝ      ｹｲﾘﾌﾞ      411
ﾔﾏｶﾜｶｽﾞｵ         38   ﾄｳｷｮｳﾄ     ｴｲｷﾞｮｳﾌﾞ   490
ｶﾜﾀﾞﾔｽｼﾞ         30   ﾁﾊﾞｹﾝ      ｴｲｷﾞｮｳﾌﾞ   450
ｴﾀﾞｺﾞﾛｳ          53   ﾄｳｷｮｳﾄ     ｴｲｷﾞｮｳﾌﾞ   1250
ｺﾔﾏｺﾞﾛｳ          46   ｱｲﾁｹﾝ      ｷｶｸﾌﾞ      850
ｲｻﾞﾜｻﾌﾞﾛｳ        50   ｺｳﾁｹﾝ      ｴｲｷﾞｮｳﾌﾞ   1000
ﾀｶﾀﾞﾋﾛｼ          53   ﾌｸｵｶｹﾝ     ｴｲｷﾞｮｳﾌﾞ   1150
ｵｵｻｷﾂﾖｼ          45   ﾎｯｶｲﾄﾞｰ    ｴｲｷﾞｮｳﾌﾞ   920
                 0                          0
                 0                          0
                 0                          0
```

(166)【プログラム例】

```
DIM a$(16, 3), x%(16, 2) ──────────── データをファイルから読み出し
OPEN "b:meibo4.dat" FOR RANDOM AS #1 LEN = 180   名前が"タカダヒロシ"のとき出身地を
FOR i = 1 TO 4                                   "カゴシマケン"に変更して配列へ代入
    FOR j = 0 TO 3
        FIELD #1, 40 * j AS ddd$, 16 AS aa$, 2 AS bb$, 10 AS cc$, 10 AS dd$, 2 AS ee$
        GET #1, i
        k = (i - 1) * 4 + j
        a$(k, 0) = aa$
        x%(k, 0) = CVI(bb$)
        a$(k, 1) = cc$
        a$(k, 2) = dd$
        x%(k, 1) = CVI(ee$)
        IF a$(k, 0) = "ﾀｶﾀﾞﾋﾛｼ          " THEN a$(k, 1) = "ｶｺﾞｼﾏｹﾝ"
    NEXT j
NEXT i
FOR i = 1 TO 4 ──────────── 配列データをファイルへかきなおす
    FOR j = 0 TO 3
        FIELD #1, 40 * j AS ddd$, 16 AS aa$, 2 AS bb$, 10 AS cc$, 10 AS dd$, 2 AS ee$
        k = (i - 1) * 4 + j
        LSET aa$ = a$(k, 0)
        LSET bb$ = MKI$(x%(k, 0))
        LSET cc$ = a$(k, 1)
        LSET dd$ = a$(k, 2)
        LSET ee$ = MKI$(x%(k, 1))
    NEXT j
PUT #1, i
NEXT i
CLOSE #1
END
```

【結　果】（ファイルの内容）

ｴﾝﾄﾞｳｲﾁﾛｳ	33	ﾎｯｶｲﾄﾞｰ	ｴｲｷﾞｮｳﾌﾞ	465
ｲｹﾀﾞｺｳｲﾁ	36	ｶｺﾞｼﾏｹﾝ	ｿｳﾑﾌﾞ	430
ﾅｶﾀﾞｺｳｼﾞ	29	ｵｳｻｶﾌ	ｷﾞｮｳﾑﾌﾞ	428
ｶｲﾀﾞﾀｲｼﾞ	43	ﾔﾏｸﾞﾁｹﾝ	ﾎﾞｳｴｷﾌﾞ	580
ｺﾝﾄﾞｳｻﾌﾞﾛｳ	25	ｱｵﾓﾘｹﾝ	ｷﾞｮｳﾑﾌﾞ	398
ｵﾀﾞﾉﾘｵ	40	ﾄﾔﾏｹﾝ	ｹｲﾘﾌﾞ	411
ﾔﾏｶﾜｶｽﾞｵ	38	ﾄｳｷｮｳﾄ	ｴｲｷﾞｮｳﾌﾞ	490
ｶﾜﾀﾞﾔｽｼﾞ	30	ﾁﾊﾞｹﾝ	ｴｲｷﾞｮｳﾌﾞ	450
ｴﾀﾞｺﾞﾛｳ	53	ﾄｳｷｮｳﾄ	ｴｲｷﾞｮｳﾌﾞ	1250
ｺﾔﾏｺﾞﾛｳ	46	ｱｲﾁｹﾝ	ｷｶｸﾌﾞ	850
ｲｻﾞﾜｻﾌﾞﾛｳ	50	ｺｳﾁｹﾝ	ｴｲｷﾞｮｳﾌﾞ	1000
ﾀｶﾀﾞﾋﾛｼ	53	ｶｺﾞｼﾏｹﾝ	ｴｲｷﾞｮｳﾌﾞ	1150
ｵｵｻｷﾂﾖｼ	45	ﾎｯｶｲﾄﾞｰ	ｴｲｷﾞｮｳﾌﾞ	920
	0			0
	0			0
	0			0

出身地が福岡県から鹿児島県に変更されている

（167）**【プログラム例】**

```
DIM a$(16, 3), x%(16, 2)
OPEN "b:meibo4.dat" FOR RANDOM AS #1 LEN = 180
FOR i = 1 TO 4
    FOR j = 0 TO 3
        FIELD #1, 40 * j AS ddd$, 16 AS aa$, 2 AS bb$, 10 AS cc$, 10 AS dd$, 2 AS ee$
        GET #1, i
        k = (i - 1) * 4 + j
        a$(k, 0) = aa$
        x%(k, 0) = CVI(bb$)
        a$(k, 1) = cc$
        a$(k, 2) = dd$
        x%(k, 1) = CVI(ee$)
        IF a$(k, 0) = "ｲｹﾀﾞｺｳｲﾁ        " THEN a$(k, 0) = "*ｲｹﾀﾞｺｳｲﾁ       "
    NEXT j
NEXT i
FOR i = 1 TO 4
    FOR j = 0 TO 3
        FIELD #1, 40 * j AS ddd$, 16 AS aa$, 2 AS bb$, 10 AS cc$, 10 AS dd$, 2 AS ee$
        k = (i - 1) * 4 + j
        LSET aa$ = a$(k, 0)
        LSET bb$ = MKI$(x%(k, 0))
        LSET cc$ = a$(k, 1)
        LSET dd$ = a$(k, 2)
        LSET ee$ = MKI$(x%(k, 1))
    NEXT j
PUT #1, i
NEXT i
CLOSE #1
END
```

ファイルからデータを読み出し，名前が”イケダコウイチ”のとき先頭に＊をつけて配列へ代入

配列データをファイルへかきなおす

【結　果】(ファイルの内容)

```
ｴﾝﾄﾞｳｲﾁﾛｳ      33
*ｲｹﾀﾞｺｳｲﾁ      36
ﾅｶﾀﾞｺｳｼﾞ       29
ｶｲﾀﾞﾀｲｼﾞ       43
ｺﾝﾄﾞｳｻﾌﾞﾛｳ     25
ｵﾀﾞﾉﾘｵ         40
ﾔﾏｶﾜｶｽﾞｵ       38
ｶﾜﾀﾞﾔｽｼﾞ       30
ｴﾀﾞｺﾞﾛｳ        53
ｺﾔﾏｺﾞﾛｳ        46
ｲｻﾞﾜｻﾌﾞﾛｳ      50
ﾀｶﾀﾞﾋﾛｼ        53
ｵｵｻｷﾂﾖｼ        45
               0
               0
               0
```

```
ﾎｯｶｲﾄﾞｰ     ｴｲｷﾞｮｳﾌﾞ    465
ｶｺﾞｼﾏｹﾝ     ｿｳﾑﾌﾞ       430
ｵｳｻｶﾌ       ｷﾞｮｳﾑﾌﾞ     428
ﾔﾏｸﾞﾁｹﾝ     ﾎﾞｳｴｷﾌﾞ     580
ｱｵﾓﾘｹﾝ      ｷﾞｮｳﾑﾌﾞ     398
ﾄﾔﾏｹﾝ       ｹｲﾘﾌﾞ       411
ﾄｳｷｮｳﾄ      ｴｲｷﾞｮｳﾌﾞ    490
ﾁﾊﾞｹﾝ       ｴｲｷﾞｮｳﾌﾞ    450
ﾄｳｷｮｳﾄ      ｴｲｷﾞｮｳﾌﾞ    1250
ｱｲﾁｹﾝ       ｷｶｸﾌﾞ       850
ｺｳﾁｹﾝ       ｴｲｷﾞｮｳﾌﾞ    1000
ｶｺﾞｼﾏｹﾝ     ｴｲｷﾞｮｳﾌﾞ    1150
ﾎｯｶｲﾄﾞｰ     ｴｲｷﾞｮｳﾌﾞ    920
                        0
                        0
                        0
```

(430 の行 —— 削除マークがついている)

22章 ランダムファイル（2）

TYPE型を使用　　　LOF

【例題159】 TYPE型を使ったランダムファイルへの書き込み

下のデータ組をランダムファイル computer.dat にかくプログラムをつくれ.

データ：	型名	社名	価格（円）
	X6800	Sharp	356000
	FM-Towns	Fujitsu	338000
	PC-9801ES/2	NEC	448000

【解　説】 ◎ TYPE型を使ってランダムファイルへデータ組を書き込みます.

① Quick BASIC ではバッファを Field 文で割りあてる従来の方法の他に TYPE 型を使って，ランダムファイルをつくることができます. この場合は 1 バッファが 256 バイトと固定した考えでなく，TYPE 型の持つバイト数で決められるためむだが少なくてすみます.

② TYPE 型でタブ computer を宣言します. 文字型 15+10 バイトと長整数 4 バイトの合計 29 バイトが 1 組の必要バイト数です.

③ ランダムファイルのオープンは次のようにします.

```
OPEN "computer.dat", FOR RANDOM AS #1 LEN=LEN(a)
```

- OPEN：オープン
- "computer.dat"：ファイル名
- FOR RANDOM：ランダムファイルを指定
- #1：オープン番号
- LEN=LEN(a)：バッファのバイト数　a は TYPE 型変数名

LEN=LEN(a)によってバッファのバイト数が決められますから，無駄なく書き込みができます.

④ 次の命令でデータ組を書き込みます.

```
PUT #1, i, a
```

- PUT：書き込み
- #1：オープン番号
- i：レコード番号　i が 1，2，3 となり 1 レコード，2 レコード，3 レコードを順にかく.
- a：書き込む内容　a.brand，a.company，a.price を 1 組として一度に書き込む

⑤ ファイルのクローズは CLOSE　#1 とオープン番号を使って行います.

【プログラム例】

```
TYPE computer ──────────────────┐─TYPE 型の宣言
     brand AS STRING * 15       │
     company AS STRING * 10     │
     price AS LONG              │
END TYPE ───────────────────────┘
DIM a  AS computer ──────────────── TYPE 型の変数 a を宣言
OPEN "b:computer.dat" FOR RANDOM AS #1 LEN = LEN(a)── ファイルのオープン
FOR i = 1 TO 3 ─────────────────┐─1 組 3 つのデータ組を 3 組読み込んで
    READ a.brand, a.company, a.price  │ レコード番号 1, 2, 3 としてファイル
    PUT #1, i, a                │ へ書き込む
NEXT i ─────────────────────────┘
CLOSE #1 ────────────────────────── ファイルのクローズ
END
DATA X6800,Sharp,356000,FM-Towns,Fujitsu,338000
DATA PC-9801ES/2,NEC,448000
```

【例題 160】　ランダムファイルの読み出し

【例題 159】で書き込んだデータを読み出して表示するプログラムをつくれ.

【解　説】　◎　ランダムファイルの読み出しをします.

① 読み出したデータを格納するタイプ型変数の宣言をします. これは書き込みした時と同じ TYPE 型とします.

② ファイルをオープンします. ファイルのオープンは書き込みと全く同じです.

③ 次の命令でデータ組を読み出します.

```
GET #1, i, a
 |    |   |  └──── タイプ型変数 3 データを 1 組として読み出す
 |    |   └レコード番号
読み出し オープン
       番号     i を 1, 2, 3 として 1 レコード, 2 レコード, 3 レコードを読み出す
```

【プログラム例】

```
TYPE computer ──────────────────┐─TYPE 型の宣言
     brand AS STRING * 15       │
     company AS STRING * 10     │
     price AS LONG ─────────────┘
END TYPE
DIM a  AS computer ──────────────── TYPE 型の変数 a を宣言
OPEN "b:computer.dat" FOR RANDOM AS #1 LEN = LEN(a)── ファイルのオープン
FOR i = 1 TO 3 ─────────────────┐─ファイルからデータを読み出して表示
    GET #1, i, a                │
    PRINT a.brand; a.company; a.price │
NEXT i ─────────────────────────┘
CLOSE #1 ────────────────────────── ファイルのクローズ
END
```

【結　果】

```
X6800          Sharp      356000
FM-Towns       Fujitsu    338000
PC-9801ES/2    NEC        448000
```

【例題 161】　ランダムファイルへ追加

【例題 159】でつくったファイル computer.dat に下のデータ組を追加して書き込むプログラムをつくれ.

データ：

型名	社名	価格
PC-286VE	Epson	298000
J-3100	Toshiba	448000

【解　説】　◎　ランダムファイルにデータ組を追加します.

① TYPE 型変数を使ってランダムファイルをつくる場合，1 組のデータが 1 レコードとして書き込まれていますから，レコード組の追加は書き込まれている最後のレコードの次に書き込めばよいことになります.

② 書き込まれているレコード数は次の式で求められます.

```
LOF(1) / LEN(a)     ……レコード数となる
  ↑         ↑
全体のバイト数  TYPE 型変数の
             総バイト数
```

③ したがって，追加でかき込むためには上の LOF(1)/LEN(a)の値+1 番目のレコードからかきこめばよいことになります.

【プログラム例】

```
TYPE computer ────────────────────── TYPE 型の宣言
     brand AS STRING * 15
     company AS STRING * 10
     price AS LONG
END TYPE
DIM a  AS computer ────────────────── TYPE 型の変数の宣言
OPEN "b:computer.dat" FOR RANDOM AS #1 LEN = LEN(a)── ファイルのオープン
n = LOF(1) / LEN(a) ───────────────── レコード数を n に求める
FOR i = n + 1 TO n + 2 ─────────────── n+1 レコード，n+2 レコードに
    READ a.brand, a.company, a.price    データを書き込む
    PUT #1, i, a
NEXT i ────────────────────────────── ファイルのクローズ
CLOSE #1
END
DATA PC-286VE,Epson,298000,J-3100,Toshiba,448000
```

【例題162】 ファイルデータの読み込み

【例題161】を行った後のファイル computer.dat のデータを読み出して表示するプログラムをつくれ.

【解　説】 ◎ ファイルデータを読み出します. レコード数の数え方を確認します.

① ファイル computer.dat の中のレコード数は不明です. レコード数は次の式で求めます.

```
LOF(1) / LEN(a)     …レコード数
  ↑         ↑
全体バイト数  TYPE型変数の
            バイト数
```

② レコード数が n 個とわかれば 1 から n までのくり返しで n レコードを順に読み出し表示します.

【プログラム例】

```
TYPE computer                                  ―TYPE型の宣言
    brand AS STRING * 15
    company AS STRING * 10
    price AS LONG
END TYPE
DIM a  AS computer                             ―TYPE型変数 a の宣言
OPEN "b:computer.dat" FOR RANDOM AS #1 LEN = LEN(a) ―ファイルのオープン
n = LOF(1) / LEN(a)                            ―レコード数を n に求める
FOR i = 1 TO n                                 ―1 から n レコードまでを
   GET #1, i, a                                  読み出して表示
   PRINT a.brand; a.company; a.price
NEXT i
CLOSE #1                                       ―ファイルのクローズ
END
```

【結　果】

```
X6800          Sharp       356000
FM-Towns       Fujitsu     338000
PC-9801ES/2    NEC         448000
PC-286VE       Epson       298000
J-3100         Toshiba     448000
```

【例題163】 ランダムファイルの書き換え

ファイル computer.dat の第2レコードを下のデータ組に入れかえるプログラムをつくれ.

データ：FM77AV40sx, Fujitsu, 267800

【解　説】 ◎ ランダムファイルの書き換えをします.

① ランダムファイルのレコードの書き換えは, 書き換えるレコード番号を指定して, 新しいデータ組を書き込めば変更が行われます.

```
PUT  #1, 2, a
 |      |    |
オープン番号 |  └── TYPE 型変数 a（新しいデータ組）
        |
   第2レコードに書き込む
```

【プログラム例】

```
TYPE computer ──────────────────┐TYPE 型の宣言
    brand AS STRING * 15        │
    company AS STRING * 10      │
    price AS LONG               │
END TYPE ───────────────────────┘
DIM a  AS computer──────────────────── TYPE 型変数 a の宣言
OPEN "b:computer.dat" FOR RANDOM AS #1 LEN = LEN(a) ── ファイルのオープン
READ a.brand, a.company, a.price ───── データの読み込み
PUT #1, 2, a ───────────────────────── 2 レコード目に書き込み
CLOSE #1 ───────────────────────────── ファイルのクローズ
END
DATA FM77AV40sx,Fujitsu,267800
```

【結　果】（ファイルの内容）

```
X6800          Sharp      356000
FM77AV40sx     Fujitsu    267800──変更データ
PC-9801ES/2    NEC        448000
PC-286VE       Epson      298000
J-3100         Toshiba    448000
```

【例題 164】　ファイルデータを配列に読み出す

ファイル computer.dat のデータ組を TYPE 型配列変数 b( ) に読み出して表示するプログラムをつくれ.

【解　説】　◎　ファイルのデータ組を配列へ読み出します.

①　ファイルのデータ組を読み出し，配列へ代入します.

②　配列を TYPE 型の配列変数とすれば次のようになります.

```
FOR i=1 TO n                    レコード数 n 個. i を 1 から n までくり返す
  GET  #1,i,  a                 TYPE 型変数 a に i 番目のレコードを読む
  b(i).brand=a.brand          ┐a.brand～a.price を b(i).brand～b(i).price
  b(i).company=a.company      │に代入
  b(i).price=a.price          ┘
NEXT i
```

③　TYPE 型配列変数ではなく 2 次元配列に代入する場合は次のように代入できます.

```
b$(i,1)=a.brand        b(i).brand のかわりに b$(i,1)を使う
b$(i,2)=a.company      b(i).company のかわりに b$(i,2)を使う
c(i)=a.price           b(i).price のかわりに c(i)を使う
                       前もって配列 DIM b$(1 TO n,1 TO 2)と DIM c(i)をしておく.
```

【プログラム例】

```
TYPE computer ─────────────────────┐TYPE 型の宣言
     brand AS STRING * 15          │
     company AS STRING * 10        │
     price AS LONG                 │
END TYPE ──────────────────────────┘
DIM a AS computer ──────────────────TYPE 型変数 a の宣言
OPEN "b:computer.dat" FOR RANDOM AS #1 LEN = LEN(a) ──ファイルのオープン
n = LOF(1) / LEN(a) ────────────────レコード数を n に求める
DIM b(1 TO n) AS computer ──────────TYPE 型配列変数 b( )を宣言
FOR i = 1 TO n ────────────────────┐ファイルのデータを読み出し
    GET #1, i, a                   │て配列変数へ代入
    b(i).brand = a.brand           │
    b(i).company = a.company       │
    b(i).price = a.price           │
NEXT i ────────────────────────────┘
CLOSE #1 ───────────────────────────ファイルのクローズ
FOR i = 1 TO n ────────────────────┐表示
    PRINT b(i).brand; b(i).company; b(i).price
NEXT i ────────────────────────────┘
END
```

【結　果】

```
X6800           Sharp       356000
FM77AV40sx      Fujitsu     267800
PC-9801ES/2     NEC         448000
PC-286VE        Epson       298000
J-3100          Toshiba     448000
```

【例題 165】　ファイルの検索

ファイル computer.dat を使って，整数 x%を入力し，x%番目のレコードのデータ組を表示するプログラムをつくれ．

【解　説】　◎　ファイルデータを検索します．

① ランダムファイルのデータの検索は，レコード番号を指定して，そのレコードのデータ組を TYPE 型変数に読み出します．

【プログラム例】

```
TYPE computer ─────────────────────┐TYPE 型の宣言
     brand AS STRING * 15          │
     company AS STRING * 10        │
     price AS LONG                 │
END TYPE ──────────────────────────┘
DIM a  AS computer ─────────────────TYPE 型変数 a の宣言
INPUT x% ───────────────────────────整数 x%の入力
OPEN "b:computer.dat" FOR RANDOM AS #1 LEN = LEN(a) ──ファイルのオープン
GET #1, x%, a ─────────────────────┐x%番目のレコードを読み出して表示
PRINT a.brand; a.company; a.price ─┘
```

```
CLOSE #1 ─────────────────────────ファイルのクローズ
END
```

【結　果】

```
? 3                                          3 を入力の例
PC-9801ES/2     NEC         448000
```

〔演習問題〕

(168)　下のデータ組を TYPE 型を使ってランダムファイル meibo5.dat にかくプログラムをつくれ.

名　前	年齢	年収
大山　太郎	38	1570
植田　義雄	41	1300
栗田　三郎	25	820
徳川　義男	52	2300
川上　太郎	21	420
天田　義満	37	1210

(169)　(168) でつくったファイル meibo5.dat のデータ組を読み出して表示するプログラムをつくれ.

(170)　(168) でつくったファイル meibo5.dat に次の 3 組のデータ組を追加して書くプログラムをつくれ.

名　前	年齢	年収
清水　　了	38	950
千葉　勝利	25	880
橋田　鉄夫	39	730

(171)　(170) でつくったファイル meibo5.dat を読み出して表示するプログラムをつくれ. ただしデータ数は不明とする.

(172)　(170) でつくったファイル meibo5.dat の第 3 レコードを下のデータ組でかきかえるプログラムをつくれ.

名　前	年齢	年収
園田　仙一	62	5230

(173)　meibo5.dat の 4 番目のデータ組を読み出して表示するプログラムをつくれ.

〔解　答〕

(168)【プログラム例】

```
TYPE meibo ─────────────────────────── TYPE 型の宣言
     namae AS STRING * 10
     nenrei AS INTEGER
     nenshu AS INTEGER
END TYPE
DIM a  AS meibo ────────────────────── TYPE 型変数 a の宣言
OPEN "b:meibo5.dat" FOR RANDOM AS #1 LEN = LEN(a) ── ファイルのオープン
FOR i = 1 TO 6 ─────────────────────── データを読んでファイルへ書き込み
    READ a.namae, a.nenrei, a.nenshu
    PUT #1, i, a
NEXT i
CLOSE #1 ───────────────────────────── ファイルのクローズ
END
DATA ｵｵﾔﾏﾀﾛｳ,38,1570,ｳｴﾀﾞ ﾖｼｵ,41,1300
DATA ｸﾘﾀｻﾌﾞ ﾛｳ,25,820,ﾄｸｶﾞ ﾜﾖｼｵ,52,2300
DATA ｶﾜｶﾐﾀﾛｳ,21,420,ｱﾏﾀﾞ ﾖｼﾐﾂ,37,1210
```

(169)【プログラム例】

```
TYPE meibo ─────────────────────────── TYPE 型の宣言
     namae AS STRING * 10
     nenrei AS INTEGER
     nenshu AS INTEGER
END TYPE
DIM a  AS meibo ────────────────────── TYPE 型変数 a の宣言
OPEN "b:meibo5.dat" FOR RANDOM AS #1 LEN = LEN(a) ── ファイルのオープン
FOR i = 1 TO 6 ─────────────────────── ファイルから 6 組のデータを読み込み
    GET #1, i, a
    PRINT a.namae; a.nenrei; a.nenshu
NEXT i
CLOSE #1 ───────────────────────────── ファイルのクローズ
END
```

【結　果】

```
ｵｵﾔﾏﾀﾛｳ     38   1570
ｳｴﾀﾞ ﾖｼｵ    41   1300
ｸﾘﾀｻﾌﾞ ﾛｳ   25   820
ﾄｸｶﾞ ﾜﾖｼｵ   52   2300
ｶﾜｶﾐﾀﾛｳ     21   420
ｱﾏﾀﾞ ﾖｼﾐﾂ   37   1210
```

(170)【プログラム例】

```
TYPE meibo ─────────────────────────────TYPE 型の宣言
     namae AS STRING * 10
     nenrei AS INTEGER
     nenshu AS INTEGER
END TYPE
DIM a  AS meibo ────────────────────────TYPE 型変数 a の宣言
OPEN "b:meibo5.dat" FOR RANDOM AS #1 LEN = LEN(a)──ファイルのオープン
n = LOF(1) / LEN(a) ────────────────────レコード数を n に求める
FOR i = n + 1 TO n + 3 ─────────────────ファイルに 3 組のレコードを追加
    READ a.namae, a.nenrei, a.nenshu
    PUT #1, i, a
NEXT i
CLOSE #1 ───────────────────────────────ファイルのクローズ
END
DATA ｼﾐｽﾞﾘｮｳ,38,950,ﾁﾊﾞｶﾂﾄｼ,25,880
DATA ﾊｼﾀﾞﾃﾂｵ,39,730
```

(171)【プログラム例】

```
TYPE meibo ─────────────────────────────TYPE 型の宣言
     namae AS STRING * 10
     nenrei AS INTEGER
     nenshu AS INTEGER
END TYPE
DIM a  AS meibo ────────────────────────TYPE 型変数 a の宣言
OPEN "b:meibo5.dat" FOR RANDOM AS #1 LEN = LEN(a)──ファイルのオープン
n = LOF(1) / LEN(a) ────────────────────レコード数を n に求める
FOR i = 1 TO n ─────────────────────────ファイルのデータを読み出して表示
    GET #1, i, a
    PRINT a.namae; a.nenrei; a.nenshu
NEXT i
CLOSE #1 ───────────────────────────────ファイルのクローズ
END
```

【結　果】

```
ｵｵﾔﾏﾀﾛｳ      38   1570
ｳｴﾀﾞﾖｼｵ      41   1300
ｸﾘﾀｻﾌﾞﾛｳ     25   820
ﾄｸｶﾞﾜﾖｼｵ     52   2300
ｶﾜｶﾐﾀﾛｳ      21   420
ｱﾏﾀﾞﾖｼﾐﾂ     37   1210
ｼﾐｽﾞﾘｮｳ      38   950
ﾁﾊﾞｶﾂﾄｼ      25   880
```

(172)【プログラム例】

```
TYPE meibo ─────────────────────────────────┐─TYPE 型の宣言
     namae AS STRING * 10                    │
     nenrei AS INTEGER                       │
     nenshu AS INTEGER                       │
END TYPE ────────────────────────────────────┘
DIM a  AS meibo ──────────────────────────────TYPE 型変数 a の宣言
READ a.namae, a.nenrei, a.nenshu──────────────データ文のデータを読む
OPEN "b:meibo5.dat" FOR RANDOM AS #1 LEN = LEN(a)──ファイルのオープン
PUT #1, 3, a ─────────────────────────────────第 3 レコード数へ書き込み
CLOSE #1 ─────────────────────────────────────ファイルのクローズ
END
DATA ｿﾉﾀﾞ ｾﾝｲﾁ,62,5230
```

【結　果】(ファイルの内容)

```
ｵｵﾔﾏﾀﾛｳ      38   1570
ｳｴﾀﾞ ﾖｼｵ     41   1300
ｿﾉﾀﾞ ｾﾝｲﾁ    62   5230      変更データ
ﾄｸｶﾞ ﾜﾖｼｵ    52   2300
ｶﾜｶﾐﾀﾛｳ      21   420
ｱﾏﾀﾞ ﾖｼﾐﾂ    37   1210
ｼﾐｽﾞ ﾘｮｳ     38   950
ﾁﾊﾞ ｶﾂﾄｼ     25   880
ﾊｼﾀﾞ ﾃﾂｵ     39   730
```

(173)【プログラム例】

```
TYPE meibo ─────────────────────────────────┐─タイプ型の宣言
     namae AS STRING * 10                    │
     nenrei AS INTEGER                       │
     nenshu AS INTEGER                       │
END TYPE ────────────────────────────────────┘
DIM a  AS meibo ──────────────────────────────配列の宣言
OPEN "b:meibo5.dat" FOR RANDOM AS #1 LEN = LEN(a)──ファイルのオープン
GET #1, 4, a ─────────────────────────────────4 番目のデータの読み出し
PRINT a.namae; a.nenrei; a.nenshu ────────────表示
CLOSE #1──────────────────────────────────────ファイルのクローズ
END
```

【結　果】

```
ﾄｸｶﾞ ﾜﾖｼｵ    52   2300
```

23章 シーケンシャルファイルを ランダムファイル化

SEEK

【例題 166】 SEEK を使ってシーケンシャルファイルをランダムファイルのように使う

下のデータをシーケンシャルファイル reidai.dat にかき，次に整数 xx%を１つ入力し，xx%番目のデータを読み出して表示するプログラムをつくれ．

データ： トウキョウ，オオサカ，タカマツ，サッポロ，チバ，センダイ，ヒロシマ，ナガサキ，トリデ，ミヤザキ

【解　説】 ◎ SEEK を使うとシーケンシャルファイルをランダムファイルのように使えます．

① SEEK は指定した位置へファイルポインタをおくことができます．したがって，その位置から読み出したり，その位置へ書き込みをしたりすることができます．

② シーケンシャルファイルは数値型，文字型に関係なく順に書き込み，読み出しができます．また，そのとき，文字型の文字の数が一定でなくてもよいという便利さがあります．

③ しかし，ファイルポインタを任意の場所におけても，特に文字型データで長さがばらばらではどこを指定してよいのかわかりません．したがって，ここでは，10 個の都市名のデータをおのおの 10 文字の長さとして格納することにします．

④ まず 10 都市名をシーケンシャルファイル reidai.dat にかきます．次がそのプログラム例です．

【プログラム例】

```
DIM x$(10)
FOR i = 0 TO 9
    READ x$(i)
NEXT i
OPEN "b:reidai.dat" FOR OUTPUT AS #1
FOR i = 0 TO 9
    WRITE #1, x$(i)
NEXT i
CLOSE #1
END
DATA "ﾄｳｷｮｳ     ","ｵｵｻｶ      "
DATA "ﾀｶﾏﾂ      ","ｻｯﾎﾟﾛ     "
DATA "ﾁﾊﾞ       ","ｾﾝﾀﾞｲ     "
DATA "ﾋﾛｼﾏ      ","ﾅｶﾞｻｷ     "
DATA "ﾄﾘﾃﾞ      ","ﾐﾔｻﾞｷ     "
```

⑤　確認のため順に読み出すと次のようになります.

【プログラム例】　読み出しのプログラム

```
OPEN "b:reidai.dat" FOR INPUT AS #1
FOR i = 0 TO 9
    INPUT #1, x$
    PRINT x$
NEXT i
CLOSE #1
END
```

【結　果】　読み出し結果

```
ﾄｳｷｮｳ
ｵｵｻｶ
ﾀｶﾏﾂ
ｻｯﾎﾟﾛ
ﾁﾊﾞ
ｾﾝﾀﾞｲ
ﾋﾛｼﾏ
ﾅｶﾞｻｷ
ﾄﾘﾃﾞ
ﾐﾔｻﾞｷ
```

⑥　次に SEEK を使ってファイルポインタを移動させます.

10 個の都市名は次のように格納されています.

```
1 2 3 4 5 6 7 8 9 1011121314151617181920212223242526272829303132333334……
      10 文字                    10 文字
" トウキョウ□□□□□"  " オオサカ□□□□□□"  " タカマツ□……
      14　桁                     14　桁
```

⑦　したがって xx%番目のデータは(xx%-1)*14+1 バイト目からかかれています. そこで, そこへファイルポインタを移します. ファイルポインタは SEEK を使って自由に動かせます. 指定位置へファイルポインタを移して, 読み出し, 書き込みします.

```
SEEK  #1,y%
  │    └───── バイト番号 (xx%番目のときは(xx%-1)*14+1 が y%の値となる)
  │
ファイルのオープン番号
```

⑧　ファイルポインタは任意の位置へ動きますからあたかもシーケンシャルファイルがランダムファイルのように扱えます.

【プログラム例】

```
INPUT xx% ─────────────────────────── xx%の入力
OPEN "b:reidai.dat" FOR INPUT AS #1 ─── ファイルのオープン
y% = (xx% - 1) * 14 + 1 ─────────────── xx%番目のデータの先頭のバイト数
SEEK #1, y% ─────────────────────────── ファイルポインタを y%へ移す
INPUT #1, x$ ────────────────────────── y%からの 1 データを読み出す
CLOSE #1 ────────────────────────────── ファイルのクローズ
PRINT xx%; "ﾊﾞﾝﾒ="; x$ ──────────────── 表示
END
```

【結　果】

```
? 3                        3を入力の例
 3 ﾊﾞﾝﾒ=ﾀｶﾏﾂ
? 4                        4を入力の例
 4 ﾊﾞﾝﾒ=ｻｯﾎﾟﾛ
? 10                       10を入力の例
 10 ﾊﾞﾝﾒ=ﾐﾔｻﾞｷ
? 6                        6を入力の例
 6 ﾊﾞﾝﾒ=ｾﾝﾀﾞｲ
```

【例題 167】　整数ファイルを seek でさがす

次のデータをシーケンシャルファイル reidai1.dat にかき，次に番号を指定して，その番号のデータを読み出して表示するプログラムをつくれ．

データ：5，3，2，7，8，9，1，5，4，9
　　　　9，4，8，2，7，4，9，1，3，0

【解　説】　◎　整数ファイルを seek を使ってさがします．

①　整数データをファイル reidai1.dat にかきます．

【プログラム例】

```
DIM x%(20)
FOR i = 0 TO 19 ──────────┐ データの読み込み
    READ x%(i)              │
NEXT i ───────────────────┘
OPEN "b:reidai1.dat" FOR OUTPUT AS #1 ─┐ データをファイルへ書き込み
FOR i = 0 TO 19                         │
    WRITE #1, x%(i)                     │
NEXT i ─────────────────────────────────┘
CLOSE #1
END
DATA 5,3,2,7,8,9,1,5,4,9
DATA 9,4,8,2,7,4,9,1,3,0
```

②　整数データの xx%番目は(xx%−1)*3+1 で先頭が求まります．

③　整数は 2 バイトそれに区切りのカンマで 3 バイトずつ使われています．

```
バイト  123  456  789  101112  ………
値       5,   3,   2,    7,    ………0
```

【プログラム例】

```
INPUT xx% ─────────────────────────── xx%番目の入力
OPEN "b:reidai1.dat" FOR INPUT AS #1 ── ファイルのオープン
SEEK #1, (xx% - 1) * 3 + 1 ─────────── ポインタを移動
INPUT #1, x% ──────────────────────── 読み出し
CLOSE #1 ──────────────────────────── ファイルのクローズ
PRINT xx%; "ﾊﾞﾝﾒ="; x% ─────────────── 表示
END
```

【結　果】

```
? 17                    17 を入力の例
 17 ﾊﾞﾝﾒ= 9
```

【例題 168】　**seek を使って浮動小数点型データを読み出す**

次のデータをファイル reidai2.dat にかき，次に xx%番目を入力して，xx%番目のデータをよみ出すプログラムをつくれ.

データ;4.3, 2.6, 3.5, 4.3, 5.4, 3.2, 2.7, 2.7, 8.9, 1.9

【解　説】　◎　浮動小数点型数値を seek を使って読み出します.

① 実数データをファイルへ書き込みます.

【プログラム例】

```
DIM x!(10)
FOR i = 0 TO 9         ─┐ データの読み込み
    READ x!(i)           │
NEXT i                 ─┘
OPEN "b:reidai2.dat" FOR OUTPUT AS #1 ─┐ データをファイルへ書き込み
FOR i = 0 TO 9                          │
    WRITE #1, x!(i)                     │
NEXT i                                 ─┘
CLOSE #1
END
DATA 4.3,2.6,3.5,4.3,5.4,3.2,2.7,2.7,8.9,1.9
```

② xx%番目のデータは次の式で先頭が計算できます.

(xx%-1)*5+1

③ 単精度実数は 4 バイト，それにカンマで，5 バイトが使われます.

バイト　1234567891011

値　　　4.3,　2.6,　3.5,　…………………1.9

【プログラム例】

```
INPUT xx%                              ── xx%番目の入力
OPEN "b:reidai2.dat" FOR INPUT AS #1   ── ファイルのオープン
SEEK #1, (xx% - 1) * 5 + 1             ── ポインタを xx%番目におく
INPUT #1, x!                           ── 読み出し
CLOSE #1                               ── ファイルのクローズ
PRINT xx%; "ﾊﾞﾝﾒ="; x!                ─┐ 表示
END                                    ─┘
```

【結　果】

```
? 3                     3 を入力の例
 3 ﾊﾞﾝﾒ= 3.5
? 10                    10 を入力の例
 10 ﾊﾞﾝﾒ= 1.9
```

【例題 169】　seek を使ってデータの修正

ファイル reidai2.dat を使い 3 番目のデータを 152.33 に変更するプログラムをつくれ.

【解　説】　◎　seek を使ってデータの修正をします.

① ファイル reidai2.dat は浮動小数点単精度型のデータですから x%番目のデータは(x%-1)*5+1 へポインタをおいて, 読み書きできます. ここでは, データを書き込みますので, 前のデータが修正されます.

【プログラム例】

```
OPEN "b:reidai2.dat" FOR APPEND AS #1———— ファイルのオープン
SEEK #1, 11 ——————————————————————————— 3 番目にポインタをおく
WRITE #1, 152.33 —————————————————————— 152.33 をかく
CLOSE #1 —————————————————————————————— ファイルのクローズ
END
```

【結　果】　(【例題 168】のプログラムで読み出して確認)

```
? 3
 3 ﾊﾞﾝﾒ= 152.33
```

〔演習問題〕

(174)　reidai.dat のファイルを使い, 6 番目を"ムロラン"に変更するプログラムをつくれ.

(175)　reidai.dat のファイルを使い, 1 番目, 3 番目, …9 番目と 1 つおきにデータを表示するプログラムをつくれ.

(176)　reidai2.dat のファイルを使い, 2 番目, 9 番目のデータの和を求めるプログラムをつくれ.

〔解　答〕

(174)**【プログラム例】**

```
OPEN "b:reidai.dat" FOR APPEND AS #1———— ファイルのオープン
SEEK #1, 71 ——————————————————————————— 6 番目へポインタをおく
WRITE #1, "ﾑﾛﾗﾝ      " ————————————————— "ムロラン"をかく
CLOSE #1 —————————————————————————————— ファイルのクローズ
END
```

【結　果】

```
? 6
 6 ﾊﾞﾝﾒ=ﾑﾛﾗﾝ
```

【例題 166】のプログラムで表示した例
6 番目が"ムロラン"となっている

(175)【プログラム例】

```
OPEN "b:reidai.dat" FOR INPUT AS #1———ファイルのオープン
FOR i% = 1 TO 9 STEP 2 ———————————————1つおきに読み出して表示
    yy% = (i% - 1) * 14 + 1 ———————————1, 3, 5…9番目へポインタをおく
    SEEK #1, yy% ——————————————————————
    INPUT #1, x$ ——————————————————————読み出し
    PRINT x$ ——————————————————————————表示
NEXT i% ———————————————————————————————
CLOSE #1 ——————————————————————————————ファイルのクローズ
END
```

【結　果】

```
トウキョウ
タカマツ
チハ゛
ヒロシマ
トリテ゛
```

(176)【プログラム例】

```
OPEN "b:reidai2.dat" FOR INPUT AS #1———ファイルのオープン
FOR i = 1 TO 2 ————————————————————————2番目と9番目のファイルデータを読んで和を求める
    READ xx% ——————————————————————————何番目を読むかデータ文で指定
    yy% = (xx% - 1) * 5 + 1 ———————————ポインタを読み出すデータへおく
    SEEK #1, yy% ——————————————————————
    INPUT #1, x! ——————————————————————データの読み出し
    xx! = xx! + x! ————————————————————和
NEXT i ————————————————————————————————
PRINT xx! —————————————————————————————表示
CLOSE #1 ——————————————————————————————ファイルのクローズ
END
DATA 2,9
```

【結　果】

```
11.5
```

24章　メタコマンド

REM　$INCLUDE, $STATIC, $DYNAMIC

【例題170】　インクルードファイル

変数piを3.14159とし，これをグローバル変数としてインクルードファイル"pai.in"で保存し，次にこれを使ってsin（20度）の値を求めるプログラムをつくれ．

【解　説】　◎　メタコマンドでインクルードファイルをとり込みます．

① GRPH，F，C，ファイル名"pai.in"，形式"I/インクルード"で初期画面としてインクルードファイル"pai.in"を次のようにつくります．

【プログラム例】

```
COMMON pi
pi = 3.14159
```

ENDをつけないこと．ENDをつけるとインクルードされたとき，そこで処理が終ってしまいます．

② 次にこれをGRPH，F，Sで保存します．

③ 次にGRPH，E，Nで新規に次のプログラムをつくります．

【プログラム例】

```
REM $INCLUDE: 'b:pai.in'———インクルードファイルのとりこみ
PRINT SIN(20 / 180 * pi)———結果の表示
END
```

④ これをGRPH，F，Aファイル名　RRR-87で格納します．

⑤ これを実行します．インクルードファイルのpiの値3.14159がインクルードされてSIN(20/180*pi)が実行され表示されます．

【結　果】

```
.3420199
```

（RRR-87を実行した結果）

【例題171】　メタコマンド　include

ファイルhai.inを使って，下のデータを配列A$(12)によみ，次にこのファイルをインクルードして1つおきに表示するプログラムをつくれ．

データ：Tokyo, Osaka, Nagoya, Fukuoka, Niigata, Sendai,
Yokohama, Kawasaki, Sapporo, Okayama, Kochi, Miyazaki

【解　説】　◎　メタコマンド$includeを使います.

① 次のファイル"hai.in"をつくり保存します.

【プログラム例】 ファイル名hai.in

```
DIM a$(12)
COMMON a$()—————————————配列のコモン宣言
FOR i = 0 TO 11—————————┐データの読み込み
    READ a$(i)——————————┘
NEXT i
DATA Tokyo,Osaka,Nagoya,Fukuoka
DATA Niigata,Sendai,Yokohama,Kawasaki
DATA Sapporo,Okayama,Kochi,Miyazaki
```

② 次の形でファイル"hai.in"をとりこみます. hai.inでは, 配列A$(12)を宣言し, それをグローバル変数としています. したがって, このプログラムの先頭でこの宣言を行ったのと同じ効果を持ちます.

```
REM  $INCLUDE:  'b:hai.in'
     メタコマンド名
     メタコマンド
  REM文のようにかく      :をいれてファイル名を''で囲っておく.
```

③ includeを使うことでインクルードファイルを共通にどのプログラムからでも使えます.〔演習問題〕(178) でもこのインクルードファイルをとりこみます.

④ インクルードファイルはプロシージャ (SUBやFUNCTION) を使えません.

⑤ インクルードファイルはテキストファイルです.

【プログラム例】

```
REM $INCLUDE: 'b:hai.in'————インクルードファイルのとりこみ
FOR i = 0 TO 10 STEP 2 ————┐結果の表示
    PRINT a$(i)            │
NEXT i ————————————————————┘
END
```

【結　果】

```
Tokyo
Nagoya
Niigata
Yokohama
Sapporo
Kochi
```

【例題 172】　インクルードファイル

次のインクルードファイルをつくり，ファイル名"kamoku.in"で保存し，次に科目番号を入力した後，その科目の 20 人の点数を入力してその表示と平均点を表示せよ．

インクルードファイル"kamoku.in"の内容

```
DIM a(20), b$(5)
COMMON a(), b$()          ファイル kamoku.in
FOR i = 1 TO 5
    READ b$(i)
NEXT i
DATA 英語,数学,国語
DATA 社会,理科
```

b$(1)に英語，b$(2)に数学，…b$(5)に理科を代入，配列 a(20)，b$(5)を宣言，それぞれをグローバルとする．

【解　説】　◎　インクルードファイル

① インクルードファイル"kamoku.in"をつくり保存します．手順は次のとおりです．

GRPH，F，C，ファイル名を b:kamoku.in，形式を I/インクルードとします．↵ で初期画面として，問題のテキストをかきます．その後 GRPH，F，S で保存します．

② 次の形で"b:kamoku.in"のインクルードファイルをとりこみます．

REM　$INCLUDE:'b:kamoku.in'

【プログラム例】

```
REM $INCLUDE: 'b:kamoku.in'―――インクルードファイルのとりこみ
INPUT "科目番号"; x ――――――番号の入力
FOR i = 1 TO 20―――――――――データの入力
    INPUT a(i)
NEXT i
PRINT b$(x); "の点数" ――――表示
FOR i = 1 TO 20
    PRINT a(i);
    t = t + a(i)
NEXT i
PRINT
PRINT "平均点"; t / 20
END
```

入力画面

```
科目番号? 2
? 54
? 89
?
```

【結　果】

```
数学の点数
 54  89  98  69  58  77  46  100  85  65  95  54  77  86  94  56  68  76  99  75
平均点 76.05
```

【例題 173】　$STATIC 配列と$DYNAMIC 配列

次のデータをダイナミック配列 a$(15)に読み，次に 1 文字を入力してそれで始まるデータを表示するプログラムをつくれ．ただし，プログラムの終りで配列を消去するものとする．

```
Shinagawa, Osaki, Gotanda, Meguro, Ebisu
Shibuya,Harajuku,Yoyogi,Shinjuku,Shinokubo
Takatanobaba,Mejiro,Ikebukuro,Otsuka,Sugamo
```

【解　説】　◎　メタコマンドによって$STATIC 配列と$DYNAMIC 配列を指定します．

① プログラムをコンパイルするときに記憶領域を確保する$STATIC 配列とプログラムを実行するとき記憶領域を確保する$DYNAMIC 配列に分けて管理することができます．

これによって，$DYNAMIC 配列を含むプログラムは実行されていないとき，メモリを他に有効に使えます．特に文字列配列，複数次元の配列はメモリ使用量が多いので有効です．

② DIM a(50)のように定数が添字の配列，添字が 10 以下で暗黙に宣言された配列は$STATIC 配列です．

③ DIM a(x)のように変数によって添字が決められる配列，最初に COMMON で宣言される配列は$DYNAMIC 配列です．

④ ここではメタコマンド$STATIC と$DYNAMIC を使って使い分けをします．

⑤ ERASE a$によって$STATIC 配列では配列の各要素の値が""（ヌル）となります．$DYNAMIC 配列では配列が削除されます．

⑥ REDIM a$(100)で$DYNAMIC 配列の割りあてのやり直しをします．添字が前と変わってもかまいません．ただし，次元数を変えることはできません．$STATIC 配列では配列が削除されたわけではありませんから再宣言はエラーとなります．

【プログラム例】

```
REM $DYNAMIC a$                                  ┐ダイナミック配列 a$(15)の宣言
DIM a$(15)                                       ┘
FOR i = 0 TO 14                                  ┐データの読み込み
    READ a$(i)                                   │
NEXT i                                           ┘
INPUT x$                                         ─1 文字の入力
FOR j = 0 TO 14                                  ┐データをよみ，x$で始まるデータを表示
    IF ASC(a$(j)) = ASC(x$) THEN PRINT a$(j)     │
NEXT j                                           ┘
ERASE a$                                         ─配列の消去
END
DATA Shinagawa,Osaki,Gotanda,Meguro,Ebisu
DATA Shibuya,Harajuku,Yoyogi,Shinjuku,Shinokubo
DATA Takatanobaba,Mejiro,Ikebukuro,Otsuka,Sugamo
```

【結　果】

```
? O―――――― 0 を入力の例
Osaki
Otsuka
? S―――――― S を入力の例
Shinagawa
Shibuya
Shinjuku
Shinokubo
Sugamo
```

〔演習問題〕

(177)　インクルードファイル pai.in を使って，cos（20度）の値を求めるプログラムをつくれ.

(178)　ファイル"hai.in"を使って，3番目のデータを表示するプログラムをつくれ.

(179)　インクルードファイル"kamoku.in"を使って，科目番号を入力し，次にその科目の20人の点数を入力して，各人の点数と最高点を表示するプログラムをつくれ.

(180)　配列 a$(6,2)を$STATIC に宣言して次のデータの中の最初の6組のデータを読み，表示した後配列を消去し，後の4データを読んで表示するプログラムをつくれ.

名　前	出身地
山内　一男	大　　阪
大友　雄次	九　　州
中山　真一	東　　北
河野　洋二	北 海 道
渋谷　　一	北　　陸
川上　三郎	東　　京
池山　　努	中　　部
名古屋泰二	四　　国
馬場　　等	東　　北
中村　　勇	東　　京

(181)　配列 a$(10,2)を$DYNAMIC 配列とし，(180)のデータ10組を読み込んだ後，配列を消去し，a$(13,2)を再宣言して再度10組のデータをよみさらに3組を入力して追加し全体を表示するプログラムをつくれ．ただし，最後に配列を消去するものとする.

〔解　答〕

(177)【プログラム例】

```
                               "pai.in"は【例題170】で作成ずみとします
REM $INCLUDE: 'b:pai.in'———インクルードファイルのとりこみ
PRINT COS(20 / 180 * pi)———結果の表示
END
```

【結　果】

```
.9396927
```

(178)【プログラム例】

```
REM $INCLUDE: 'b:hai.in'———インクルードファイルのとりこみ
PRINT a$(3)———————————表示
END
```

【結　果】

```
Fukuoka
```

(179)【プログラム例】

```
REM $INCLUDE: 'b:kamoku.in'——————インクルードファイルのとりこみ
max = 0 ————————————————————————————max を 0 とする.
INPUT "科目番号"; x ————————————————科目番号の入力
FOR i = 1 TO 20 ————————————————┐点数の入力と最高点を max に求める
    INPUT a(i)                   │
    IF max < a(i) THEN max = a(i)│
NEXT i ——————————————————————————┘
PRINT b$(x); "の点数"————————————┐20人分の点の表示
FOR i = 1 TO 20                  │
    PRINT a(i);                  │
NEXT i ——————————————————————————┘
PRINT
PRINT "最高点"; max ————————————————最高点の表示
END
```

入力画面

```
科目番号? 5
? 85
? 65
?
```

【結　果】

```
理科の点数
 85  65  28  95  46  78  55  87  91  84  66  75  29  97  75  64  58  84  70  90

最高点 97
```

(180)【プログラム例】

```
REM $STATIC a$ ──────────┐── スタティック配列a$(6,2)の宣言
DIM a$(6, 2) ────────────┘
FOR i = 0 TO 5 ──────────┐── 配列へデータを6個読み込む
    READ a$(i, 0), a$(i, 1)
NEXT i ──────────────────┘
FOR j = 0 TO 5 ──────────┐── 6個のデータの表示
    PRINT a$(j, 0); "  "; a$(j, 1)
NEXT j ──────────────────┘
ERASE a$ ──────────────────── 配列の初期化
RESTORE 100 ───────────────── データポインタを移す
FOR k = 0 TO 3 ──────────┐── 後半4つのデータ組を読み込む
    READ a$(k, 0), a$(k, 1)
NEXT k ──────────────────┘
FOR i = 0 TO 3 ──────────┐── 4データ組の表示
    PRINT a$(i, 0); "  "; a$(i, 1)
NEXT i ──────────────────┘
END
DATA ﾔﾏｳﾁ ｶｽﾞｵ,ｵｵｻｶ,ｵｵﾄﾓ ﾕｳｼﾞ,ｷｭｳｼｭｳ,ﾅｶﾔﾏｼﾝｲﾁ,ﾄｳﾎｸ
DATA ｺｳﾉ ﾖｳｼﾞ,ﾎｯｶｲﾄﾞｰ,ｼﾌﾞﾔ ﾊｼﾞﾒ,ﾎｸﾘｸ,ｶﾜｶﾐ ｻﾌﾞﾛｳ,ﾄｳｷｮｳ
100 DATA ｲｹﾔﾏ ﾂﾄﾑ,ﾁｭｳﾌﾞ,ﾅｺﾞﾔ ﾀｲｼﾞ,ｼｺｸ,ﾊﾞﾊﾞ ﾋﾄｼ,ﾄｳﾎｸ,ﾅｶﾑﾗ ｲｻﾑ,ﾄｳｷｮｳ
```

【結　果】

```
ﾔﾏｳﾁ ｶｽﾞｵ   ｵｵｻｶ
ｵｵﾄﾓ ﾕｳｼﾞ   ｷｭｳｼｭｳ
ﾅｶﾔﾏ ｼﾝｲﾁ   ﾄｳﾎｸ
ｺｳﾉ ﾖｳｼﾞ   ﾎｯｶｲﾄﾞｰ
ｼﾌﾞﾔ ﾊｼﾞﾒ   ﾎｸﾘｸ
ｶﾜｶﾐ ｻﾌﾞﾛｳ   ﾄｳｷｮｳ
ｲｹﾔﾏ ﾂﾄﾑ   ﾁｭｳﾌﾞ
ﾅｺﾟﾔ ﾀｲｼﾞ   ｼｺｸ
ﾊﾞﾊﾞ ﾋﾄｼ   ﾄｳﾎｸ
ﾅｶﾑﾗ ｲｻﾑ   ﾄｳｷｮｳ
```

(181)【プログラム例】

```
REM $DYNAMIC a$ ─────────┐── ダイナミック配列a$(10,2)の宣言
DIM a$(10, 2) ───────────┘
FOR i = 0 TO 9 ──────────┐── データを読み込む
    READ a$(i, 0), a$(i, 1)
NEXT i ──────────────────┘
ERASE a$ ──────────────────── 配列の消去
REDIM a$(13, 2) ───────────── 配列の再宣言
RESTORE ───────────────────── データポインタをもどす
FOR j = 0 TO 9 ──────────┐── データを読み込む
    READ a$(j, 0), a$(j, 1)
NEXT j ──────────────────┘
FOR k = 10 TO 12 ────────┐── 10～12番目のデータを入力する
    INPUT a$(k, 0)
    INPUT a$(k, 1)
NEXT k ──────────────────┘
```

```
FOR i = 0 TO 12 ──────────────┐─0～12番目のデータを表示
    PRINT a$(i, 0); "  "; a$(i, 1)  │
NEXT i ────────────────────────┘
ERASE a$───────────────────────────配列の消去
END
DATA ﾔﾏｳﾁ ｶｽﾞｵ,ｵｵｻｶ,ｵｵﾄﾓ ﾕｳｼﾞ,ｷｭｳｼｭｳ,ﾅｶﾔﾏｼﾝｲﾁ,ﾄｳﾎｸ
DATA ｺｳﾉ ﾖｳｼﾞ,ﾎｯｶｲﾄﾞｰ,ｼﾌﾞﾔ ﾊｼﾞﾒ,ﾎｸﾘｸ,ｶﾜｶﾐ ｻﾌﾞﾛｳ,ﾄｳｷｮｳ
100 DATA ｲｹﾔﾏ ﾂﾄﾑ,ﾁｭｳﾌﾞ,ﾅｺﾞﾔ ﾀｲｼﾞ,ｼｺｸ,ﾊﾞﾊﾞ ﾋﾄｼ,ﾄｳﾎｸ,ﾅｶﾑﾗ ｲｻﾑ,ﾄｳｷｮｳ
```

【結　果】

```
ヤマウチ カス゛オ   オオサカ
オオトモ ユウシ゛   キュウシュウ       "ウチダキヨシ"以下"トウホク"までを入力の例
ナカヤマ シンイチ   トウホク
コウノ ヨウシ゛    ホッカイト゛ー
シフ゛ヤ ハシ゛メ   ホクリク
カワカミ サフ゛ロウ  トウキヨウ
イケヤマ ツトム    チュウフ゛
ナコ゜ヤ タイシ゛   シコク
ハ゛ハ゛ ヒトシ    トウホク
ナカムラ イサム    トウキヨウ
ウチタ゛ キヨシ    シコク
ノタ゛ ヒロシ     ホッカイト゛ー
スカ゛ クニオ     トウホク
```

25章　再帰プログラム

【例題 174】　再帰プログラム

整数を入力し，その値の階乗を求めるプログラムをつくれ．

ただし，階乗を求める部分を FUNCTION とし，再帰プログラムを使うものとする．

【解　説】　◎　再帰プログラムをつくります．

① 再帰プログラムとはある処理の中で自分自身を呼び出してくり返していくプログラムです．当然終了の条件をうまくつくってやる必要があります．

② 再帰の考え方の例として階乗を考えます．4 の階乗(4!)(!は階乗の意味，BASIC の単精度型実数を意味する宣言文字ではありません）は 4*3*2*1 です．したがって 4*3!となります．3!は 3*2!となります．2!は 2*1!となります．

③ 一般的に n の階乗は n*(n-1)!　で求められることがわかります．

④ n!を求めるために(n-1)の階乗を求めるプロシジュアを使い，(n-1)!を求めるために(n-2)!を求めるプロシジュアを使います．このように階乗を求めるプロシジュアを自分自身の中で呼んで使います．

⑤ 次の形で再帰を使います．

```
FUNCTION kaijou  (s%)
        IF   S%=0 THEN
                kaijou=1
        ELSE
                kaijou=s%*kaijou(s%-1)
        ENDIF
END FUNCTION
```

入力された値が S%に引きわたされます．例として 4 とします．最初は S%が 0 ではありませんから kaijou=4*kaijou(4-1) となります．

ここで kaijou(4-1)が呼び出されます．S%が 3 として呼び出されます．最初は S%は 0 ではありませんから kaijou=3*kaijou(3-1)です．

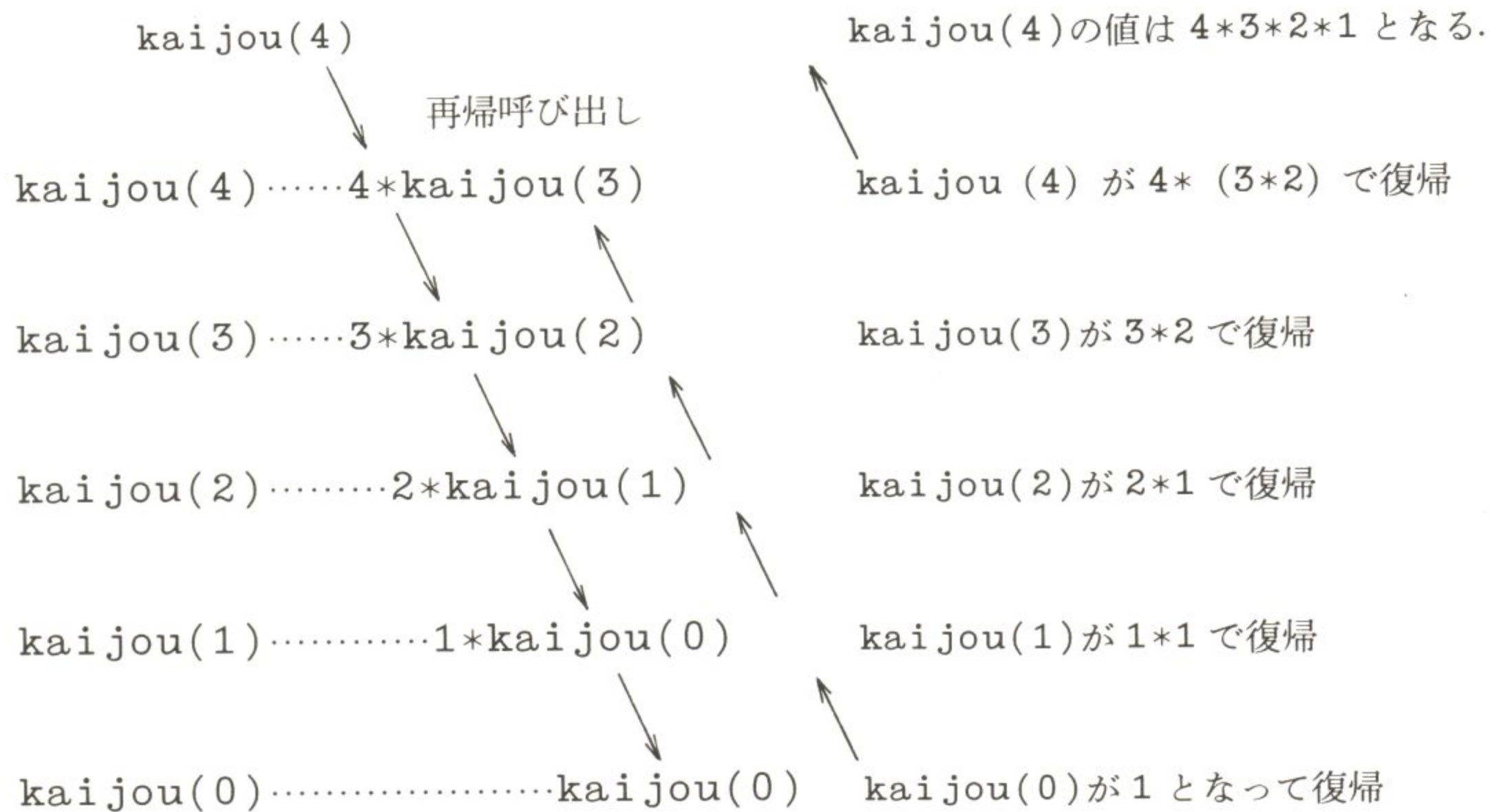
ここで kaijou(3-1)が呼び出されます．S%が 2 として呼び出されます．最初は S%は 0 ではありませんから kaijou=2*kaijou(2-1)です．

ここで kaijou(2-1)が呼び出されます．S%が 1 として呼び出されます．最初は S%は 0 ではありませんから kaijou=1*kaijou(1-1)です．

ここで kaijou(1-1)が呼び出されます．S%が 0 として呼び出されます．S%が 0 ですから，

kaijouの値は1としてプロシジュアから復帰します.

すなわちkaijou(1-1)は1です．したがってkaijou=1*kaijou(1-1)は1となります．この値で復帰します．するとkaijou(2-1)は1ですからkaijou=2*kaijou(2-1)によりこれは2です．これで復帰するとkaijou=3*kaijou(3-1)によりこの値は3*2で6です．したがってこれで復帰するとkaijou=4*kaijou(4-1)により，この値は4*6で24です．この値がkaijouの値となりメインにもどります.

```
kaijou(4)                                  kaijou(4)の値は4*3*2*1となる.
            再帰呼び出し
kaijou(4)……4*kaijou(3)                     kaijou (4) が4* (3*2) で復帰
kaijou(3)……3*kaijou(2)                     kaijou(3)が3*2で復帰
kaijou(2)………2*kaijou(1)                   kaijou(2)が2*1で復帰
kaijou(1)…………1*kaijou(0)                 kaijou(1)が1*1で復帰
kaijou(0)…………………kaijou(0)              kaijou(0)が1となって復帰
```

【プログラム例】

```
DECLARE FUNCTION kaijou (s%)
INPUT seisu% ──────────────────────整数の入力
PRINT seisu%; "!="; kaijou(seisu%)───────kaijouを呼び出し結果を表示
END

FUNCTION kaijou (s%) ──────────────階乗を求めるFUNCTION
    IF s% = 0 THEN ──────────────┬─S%が0のときkaijouを1として終了
        kaijou = 1───────────────┘
    ELSE ────────────────────────┬─kaijouの再帰呼び出し
        kaijou = s% * kaijou(s% - 1)─┘
    END IF
END FUNCTION
```

【結　果】

```
? 5                     5を入力の例
 5 != 120

? 10                    10を入力の例
 10 != 3628800

? 15                    15を入力の例
 15 != 1.307674E+12
```

〔演習問題〕

(182)　整数を１つ入力して，１からその値まで和を求めるプログラムをつくれ．

(183)　整数を２つ入力して，整数１の中から整数２をとり出す順列を求めるプログラムをつくれ．ただし，順列は整数１の階乗/(整数１-整数２)の階乗である．

(184)　整数を２つ入力して，整数１の中から整数２をとり出す組合わせの数を求めるプログラムをつくれ．ただし，組合わせ数は整数１の階乗/（整数１-整数２）の階乗/整数２の階乗である．

〔解　答〕

(182)【プログラム例】

```
DECLARE FUNCTION wa (s%)
INPUT seisu% ──────────────────────────────整数の入力
PRINT "1 ｶﾗ "; seisu%; "ﾏﾃﾞ ﾉ ﾜ ="; wa(seisu%) ───表示，wa を呼び出す
END

FUNCTION wa (s%) ───────────────────────────FUNCTION wa
    IF s% = 0 THEN ─────────────────────────┐終了条件
        wa = 0 ─────────────────────────────┘
    ELSE
        wa = s% + wa(s% - 1)────────────────────再帰呼び出し
    END IF
END FUNCTION
```

【結　果】

```
? 10                          10 を入力の例
1 ｶﾗ  10 ﾏﾃﾞﾉ ﾜ = 55
? 25                          25 を入力の例
1 ｶﾗ  25 ﾏﾃﾞﾉ ﾜ = 325
```

(183)【プログラム例】

```
DECLARE FUNCTION junretsu (s%)
INPUT seisu1% ─────────────────────────┐整数を２つ入力
INPUT seisu2% ─────────────────────────┘
ku1% = junretsu(seisu1%) ───────────────整数 seisu1 の階乗
ku2% = junretsu(seisu1% - seisu2%)──────整数 1-整数 seisu 2 の階乗
PRINT seisu1%; "ﾉ ﾅｶｶﾗ "; seisu2% ──────┐結果の表示
PRINT "ｵ ﾄﾘﾀﾞｽ ｼﾞｭﾝﾚﾂ ="; ku1% / ku2%───┘
END

FUNCTION junretsu (s%) ─────────────────階乗を求める FUNCTION
    IF s% = 0 THEN─────────────────────┐終了条件
        junretsu = 1───────────────────┘
```

```
    ELSE
        junretsu = s% * junretsu(s% - 1) ―――再帰呼び出し
    END IF
END FUNCTION
```

【結　果】

```
? 3                                      3と2を入力の例
? 2
 3 ノ ナカカラ  2 オ トリダ゛ス ジ゛ュンレツ = 6
? 5                                      5と3を入力の例
? 3
 5 ノ ナカカラ  3 オ トリダ゛ス ジ゛ュンレツ = 60
? 7                                      7と4を入力の例
? 4
 7 ノ ナカカラ  4 オ トリダ゛ス ジ゛ュンレツ = 840
? 10                                     10と8を入力の例
? 8
 10 ノ ナカカラ  8 オ トリダ゛ス ジ゛ュンレツ = 12160
```

(184)【プログラム例】

```
DECLARE FUNCTION kumiawase (s%)
INPUT seisu1% ――――――――――――――┐整数を2つ入力
INPUT seisu2% ――――――――――――――┘
ku1% = kumiawase(seisu1%) ――――――――――――整数1の階乗
ku2% = kumiawase(seisu1% - seisu2%) ―――――(整数1－整数2)の階乗
ku3% = kumiawase(seisu2%) ――――――――――――整数2の階乗
PRINT seisu1%; "ﾉ ﾅｶｶﾗ "; seisu2%―――――┐結果の表示
PRINT "ｵ ﾄﾘﾀﾞｽ ｸﾐｱﾜｾ ="; ku1% / ku2% / ku3%―┘
END

FUNCTION kumiawase (s%) ―――――――――――――階乗を求める FUNCTION
    IF s% = 0 THEN ――――――――――――――――――終了条件
        kumiawase = 1 ―――――――――――――――再帰呼び出し
    ELSE
        kumiawase = s% * kumiawase(s% - 1)
    END IF
END FUNCTION
```

【結　果】

```
? 7                                  7と2を入力の例
? 2
 7 ノ ナカカラ  2 オ トリダ゛ス クミアワセ = 21
? 6                                  6と3を入力の例
? 3
 6 ノ ナカカラ  3 オ トリダ゛ス クミアワセ = 20
```

26章　バージョン4.5で追加された命令

SOUND, SLEEP, ストップ・キー割込み, ヘルプ・キー割込み

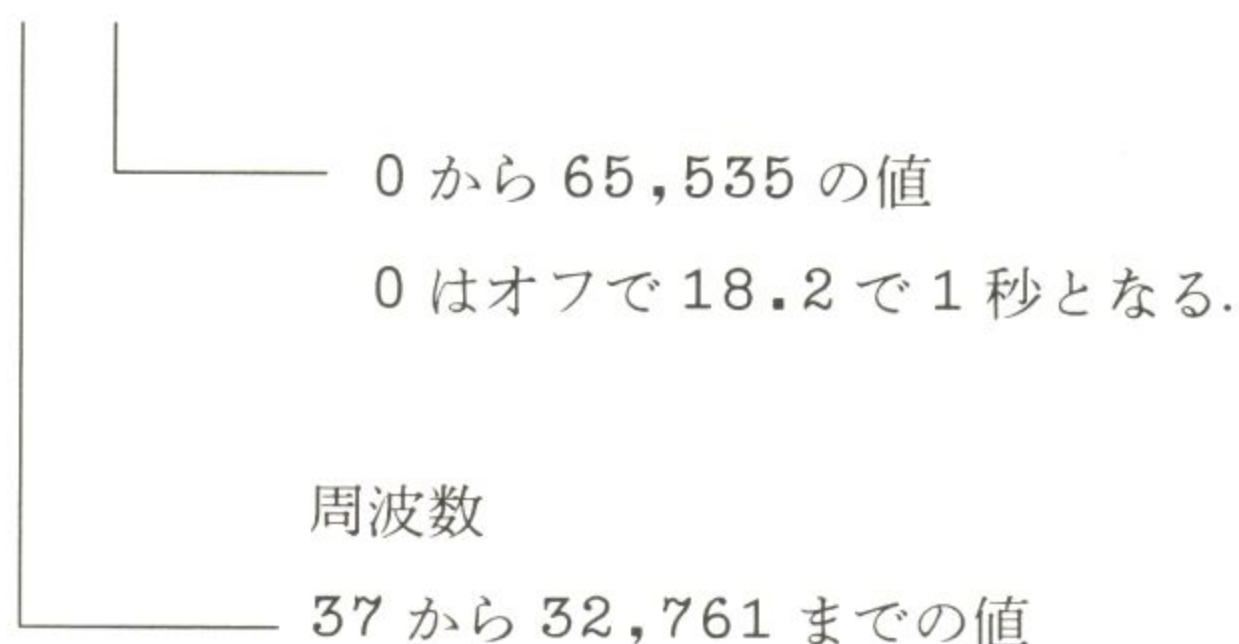
【例題175】　音（Ver. 4.5以降の機能）

ドレミファソラシド　を鳴らすプログラムをつくれ.

【解　説】　◎　SOUNDは指定した周波数の音を指定した長さだけ鳴らします.

①　SOUND x, y　は周波数xの音をy時間の間だけ鳴らします.

SUUND x, y

- y: 0から65,535の値
  0はオフで18.2で1秒となる.
- x: 周波数
  37から32,761までの値

②　音階をつくってみます.

ドレミファソラシドは下のドを1, 上のドを2としてその間を下のように分けたものです. 下のドは周波数で約520ヘルツです. したがって, xを520, yを9.1とすると下のドを0.5秒鳴らします.

ド	レ	ミ	ファ	ソ	ラ	シ	ド
1	$\sqrt[6]{2}$	$\sqrt[3]{2}$	$\sqrt[12]{2^5}$	$\sqrt[12]{2^7}$	$\sqrt[4]{2^3}$	$\sqrt[12]{2^{11}}$	2

③　この機能は9801E/F/M/Uでは使えません.

【プログラム例】

```
xx = 520: y = 9.1
FOR i = 1 TO 8
     READ x ―――――――――――――――― 音階を示す
     SOUND x * xx, y ――――――――― 周波数と0.5秒を示す.
NEXT i
END
DATA 1,1.1225,1.2599,1.3348――― 音階データ
DATA 1.4983,1.6818,1.8877,2
```

【結　果】

ドラミファソラシドが鳴ります.

> **【例題 176】　1 時停止（Ver. 4.5 以降の機能）**
>
> "1 ビョウゴニヒョウジ"と表示して，1 秒後に 1，"2 ビョウゴニヒョウジ"と表示して 2 秒後に 2 を表示，以下同様に"5 ビョウゴニヒョウジ"と表示して 5 秒後に 5 と表示するプログラムをつくれ.

【解　説】　◎　SLEEP は 1 時停止します.

① SLEEP x　は x 秒間コンピュータを一時停止します．再開する条件は次のとおりです．x が省略されると再開の条件は次の（2）と（3）です.

（1）x 秒の経過.

（2）何かキーがおされたとき.

（3）再開の割込み処理が行われたとき.

【プログラム例】

```
CLS
FOR i = 1 TO 5 ─────────────────── 1～5 のくり返し
    PRINT i; "ビョウ ゴニ ヒョウジ" ─── i ビョウゴ=ヒョウジ の表示
    SLEEP i ────────────────────── i 秒停止
    PRINT i ────────────────────── i の表示
NEXT i
END
```

【結　果】

```
1 ビョウ ゴニ ヒョウジ
1
2 ビョウ ゴニ ヒョウジ
2
3 ビョウ ゴニ ヒョウジ
3
4 ビョウ ゴニ ヒョウジ
4
5 ビョウ ゴニ ヒョウジ
5
```

> **【例題 177】　ストップキー割込み（Ver. 4.5 以降の機能）**
>
> 1～1000 のくり返しを FOR～NEXT 文でかき，その間にストップキーがおされたらその時の i を表示するプログラムをつくれ.

【解　説】　◎　ストップキーによる割込みをします.

① 次の形でストップキー割込みをします.

```
  ┌ ストップ・キーを示す  ┌ 割込み処理サブルーチン名
  │
ON KEY(30)  GOSUB  ラベル ─────── 割込み
KEY(30)  ON ──────────────────── 割込みオン
〜
```

```
END
ラベル；　──────── 割込み処理ルーチン
 ≀
RETURN
```

【プログラム例】

```
ON KEY(30) GOSUB stopkey──── ストップ・キー割込み
KEY(30) ON ────────────── 割込みオン
CLS
FOR i = 1 TO 10000: NEXT i─── くり返し
END
stopkey: ──────────────── 割込み処理ルーチン
PRINT i;
RETURN
```

【結　果】

```
2832   5086   8077
```

途中で３回 STOP キーをおした例

> 【例題 178】　**ヘルプキー割込み（Ver. 4.5 以降の機能）**
>
> 1〜1000 のくり返しを FOR〜NEXT 文でかき，その間にヘルプキーがおされたら，そのときの値とその平方根を表示するプログラムをつくれ．

【解　説】　◎　ヘルプキーによる割込みをします．

① 次の形でヘルプ・キー割込みをします．

```
          ヘルプ・キーを示す    ┌ 割込み処理サブルーチン名
ON KEY(31)  GOSUB  ラベル ──────── 割込み
KEY(31)  ON ───────────────────── 割込みオン
 ≀
END
ラベル；　──────── 割込み処理ルーチン
 ≀
RETURN
```

【プログラム例】

```
ON KEY(31) GOSUB helpkey──── ヘルプキー割込み
KEY(31) ON ────────────── 割込みオン
CLS
FOR i = 1 TO 10000: NEXT i─── くり返し
END
helpkey: ──────────────── 割込み処理
PRINT i; SQR(i);
RETURN
```

【結　果】

```
2683  51.79768  5033  70.94364  9014  94.94209
```

３回 HELP キーをおした例．

III……………グラフィック編

バージョン 4.5 では環境変数 QBGRPNEC=YES のセット (A>SET QBGRPNEC=YES)により，グラフィック画面は MS-DOS にもどしてもクリアしません.

バージョン 4.2 ではクリアしますのでグラフィック画面は MS-DOS にもどすときに消えてしまいます.

1章　直線，四角形，円，楕円，扇形，点

SCREEN, CLS, LINE, LINE STEP, LINE～B (F) CIRCLE, PSET, PRESET

【例題1】　直線の表示

画面左上から右下へ斜め斜線をかくプログラムをつくれ.

【解　説】　◎　直線をかきます.

① SCREEN 文は画面のモードを設定します.

② SCREEN　0 または SCREEN　88 は横 640 ドット*縦 400 ドットのグラフィック画面とテキスト画面を 2 頁用意します. (PC9801/U の場合は 1 頁) モードは次のようなものがあります.

モード	種　類	グラフィック解像度
81	テキスト	(80 桁*25 桁)
84	スーパーインポーズ	640*200
87	グラフィック	640*400
88, 0	スーパーインポーズ	640*400

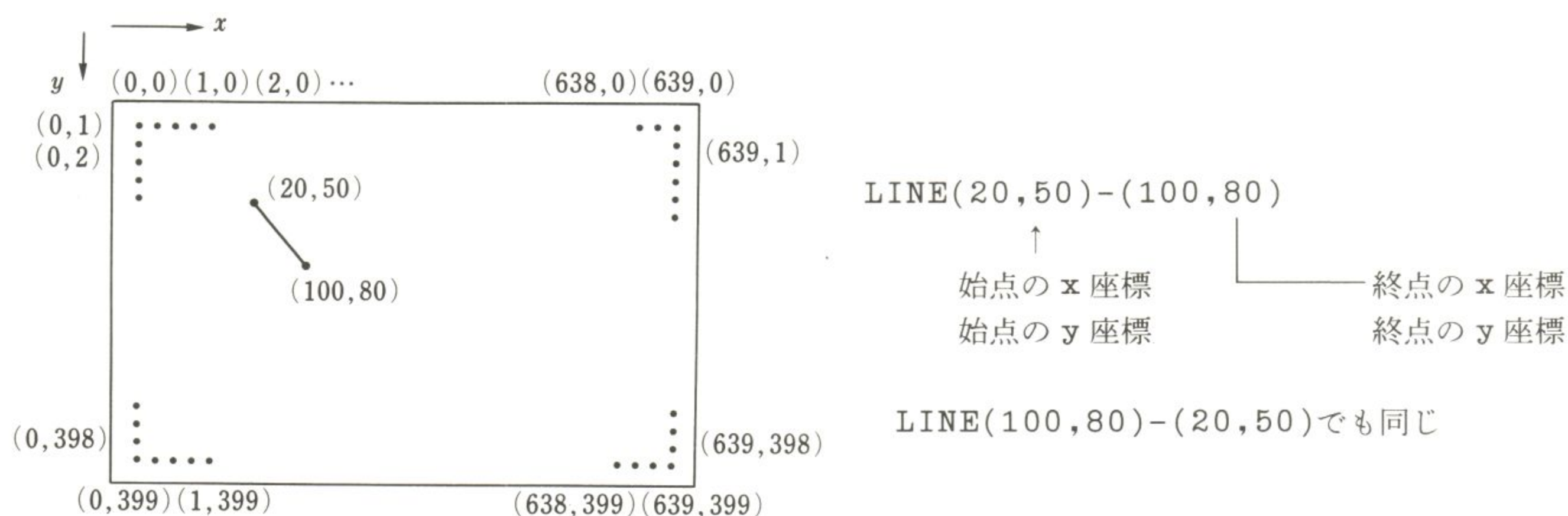
③ CLS　0 はグラフィック画面をクリアーします.

④ 直線は下図の座標系の中で指定された 2 つの位置を直線でむすびます.

→ x
↓ y
(0,0)(1,0)(2,0)… (638,0)(639,0)
(0,1)
(0,2)
(639,1)
(20,50)
(100,80)
(0,398)
(639,398)
(0,399)(1,399) (638,399)(639,399)

LINE(20,50)-(100,80)

始点の x 座標
始点の y 座標
終点の x 座標
終点の y 座標

LINE(100,80)-(20,50)でも同じ

⑤ グラフィックポインタがかき終った (100, 80) におかれます.

【プログラム例】

```
SCREEN 0 ———————— 画面設定
CLS 0 ————————— クリア
LINE (20, 50)-(100, 80)——— 直線
END
```

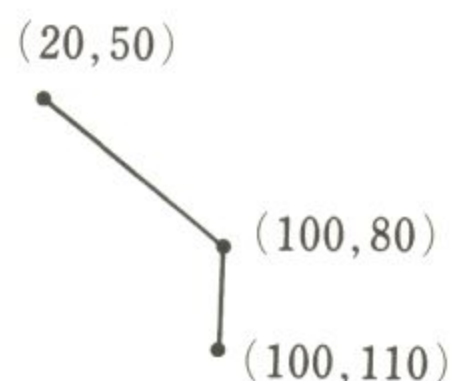
【結　果】

【例題2】　連続した直線

(20,50)と(100,80)と(100,110)をむすび, 次に(100,110)から(-20,+30)の点とそこから(+50,+20)の点をむすび, 最後にそこから(+50,0)の点をむすぶ図をつくるプログラムをつくれ.

【解　説】 ◎ 連続した直線をかきます. 絶対座標と相対座標の表わし方を示します.

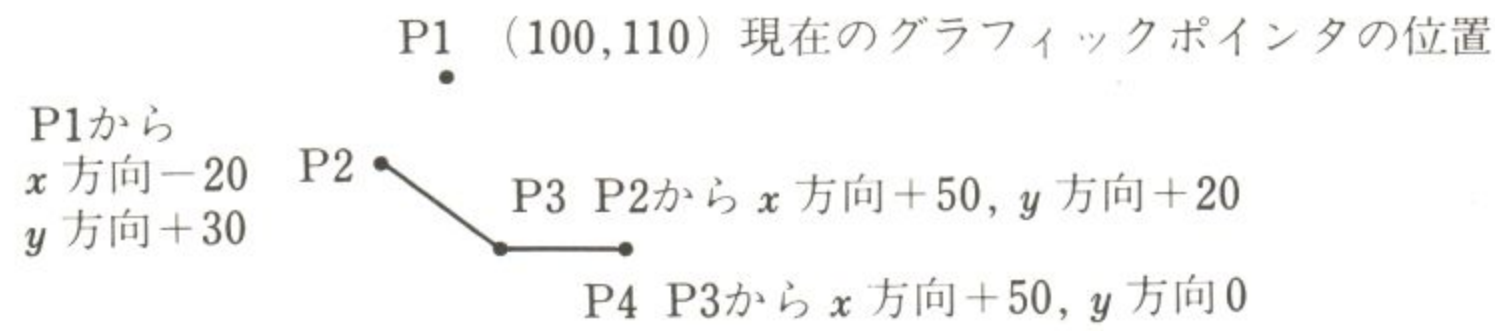
① 次の形で (20, 50) と (100, 80) および (100, 110) を直線で結びます. 各々の点の座標は絶対座標値で表わされています.

LINE(20,50)-(100,80)	(20,50)と(100,80)を直線で結ぶ
LINE-(100,110)	現在グラフィックポインタのある (100, 80) と (100,110) を結ぶ

(20,50)
(100,80)
(100,110)

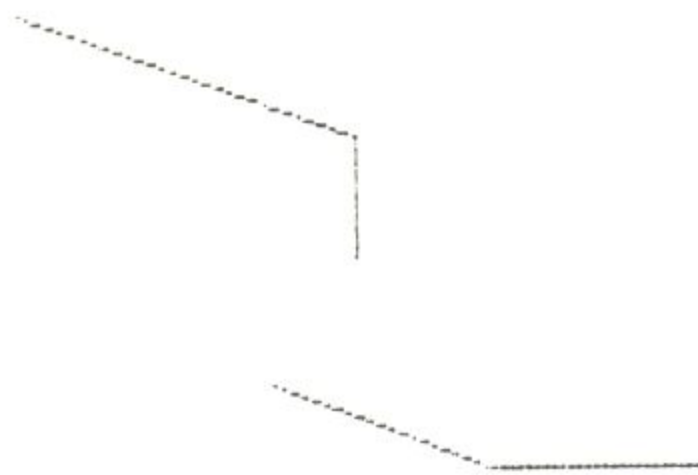
② 現在のグラフィックポインタを基準にして相対値を与えて直線をひくことができます.

LINE STEP(-20,30) -STEP(50,20) ——— 現在のグラフィックポインタから x 方向-20, y 方向+30 の位置と, そこから x 方向+50, y 方向+20 の位置をむすぶ.

LINE-STEP(50,0) ———— さらにそこから x 方向+50，y 方向 0 の位置をむすぶ

P1 (100,110) 現在のグラフィックポインタの位置
P1から x 方向－20 y 方向＋30　P2
P3 P2から x 方向＋50，y 方向＋20
P4 P3から x 方向＋50，y 方向0

【プログラム例】

```
SCREEN 0
CLS 0
LINE (20, 50)-(100, 80)
LINE -(100, 110)
LINE STEP(-20, 30)-STEP(50, 20)
LINE -STEP(50, 0)
END
```

【結　果】

【例題 3】　箱

四角形と，中を塗りつぶした四角形をかくプログラムをつくれ.

【解　説】　◎　箱と中を塗りつぶした四角形をかきます.

① 次の形で箱（四角形）をかきます.

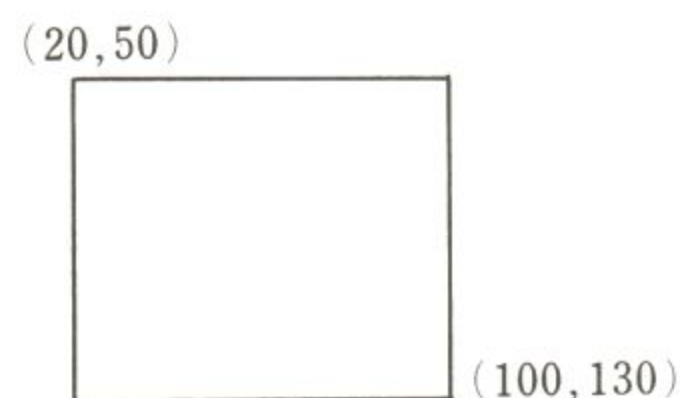
LINE(20,50)-(100,80),,B

(20,50)　(100,80)

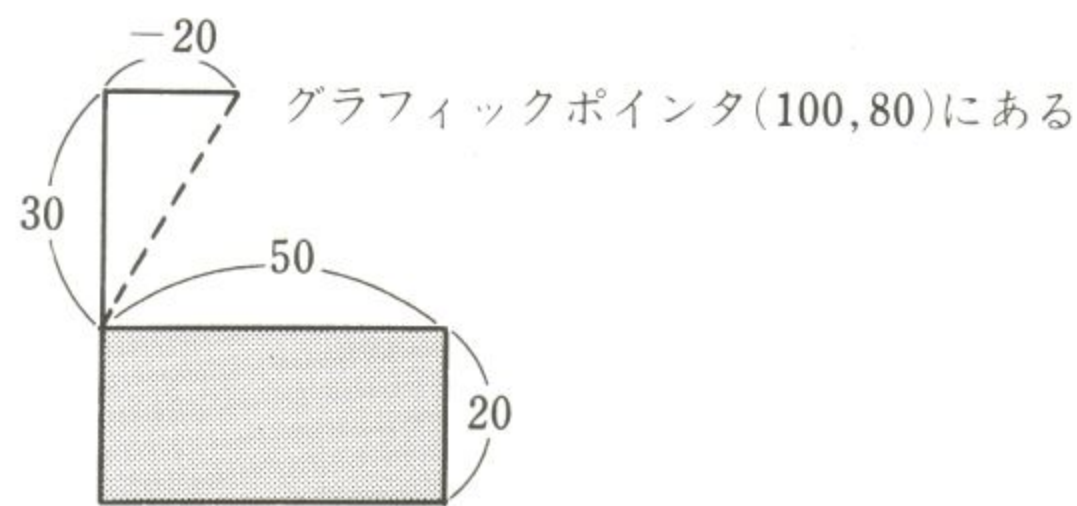
LINE(20,50)-(100,80),,B
(20,50)と(100,80)を対角とする四角. グラフィックポインタは(100,80)におかれます.
LINE(100,80)-(20,50),,B でも同じ
LINE(100, 50)-(20, 80),,B でも同じ
LINE(20, 80)-(100, 50),,B でも同じ
} グラフィックポインタは後の点となります.

② x 方向の増分と y 方向の増分が同じとき正方向となります.

LINE(20,50)-(100,130),,B は正方形となります.

(20,50)　(100,130)

③　箱の場合も STEP を使って，位置を相対的に与えることができます．また B を BF とすると箱の中を塗りつぶします．

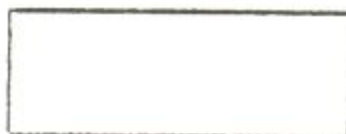
```
LINE STEP(-20,30)-STEP(50,20),,BF
```

−20　グラフィックポインタ(100,80)にある
30　50　20

BFで中を塗りつぶす

(100,80)から x 方向-20，y 方向+30 の点とそこから x 方向+50，y 方向+20 の点を対角とする箱をかき中を塗ります

【プログラム例】

```
SCREEN 0
CLS 0
LINE (20, 50)-(100, 80), , B
LINE STEP(-20, 30)-STEP(50, 20), , BF
END
```

【結　果】

【例題 4】　ラインの指定

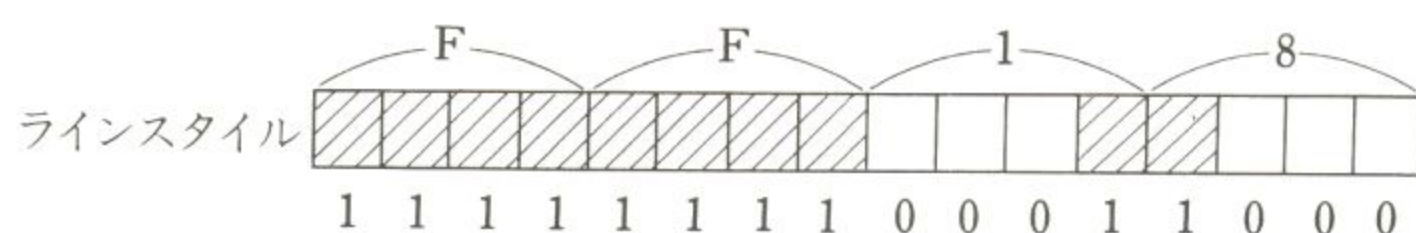
点線と 1 点鎖線で四角形をかくプログラムをつくれ．

【解　説】　◎　ラインオプションで線の種類を指定できます．

①　LINE 文の 3 番目のオプションにラインスタイルをつけることができます．ラインスタイルは次のように決めます．

```
LINE(20,100)-(100,130),,B,&HFF18 の例
```

ラインスタイル　F　F　1　8
1 1 1 1 1 1 1 1 0 0 0 1 1 0 0 0

図

16 ビットを 4 ビットずつに区切って 16 進数で表わす．この例では & HFF18 となる．このパターンのくり返しとなる．したがって 1 点鎖線となる．

②　点線は次のようになります．

点線の例

F　O　F　O　大きい点線

C　C　C　C　中位の点線

A　A　A　A　細かい点線

【プログラム例】

```
SCREEN 0
CLS 0
LINE (20, 50)-(100, 80), , B, &HF0F0
LINE (20, 100)-(100, 130), , B, &HFF18
END
```

【結　果】

【例題5】　円

中心を（100，100）とし，半径を40とする円をかくプログラムをつくれ．

【解　説】　◎　円をかきます．

①　円は次のようにかきます．

```
CIRCLE(100,100),40
```

円　中心のX座標　中心のY座標　半径

【プログラム例】

```
SCREEN 0
CLS 0
CIRCLE (100, 100), 40
END
```

【結　果】

【例題6】　半円と中心位置を相対指定

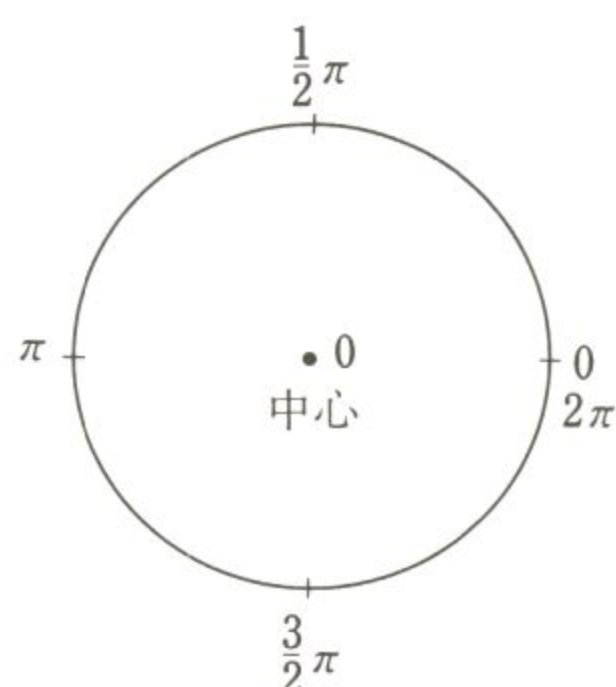
中心を（100，100），半径を 40 とする上半分の半円と，中心を x 方向に+200 した位置に半径 50 の円をかくプログラムをつくれ.

【解　説】　◎　半円と中心を相対指定します.

①　次の形で半円をかきます.

```
CIRCLE(100, 100),40,,0,3.14159
```

中心座標　半径　開始点　終了点

0 から π まで反時計方向でかく

開始点と終了点の表わし方
0～2 ラジアンで表わす.
表示は反時計方向

②　中心座標は相対的に表わすことができます.

```
CIRCLE STEP(200,0),50
```

グラフィックポインタ（本例では 100，100）から x 方向に+200，y 方向は 0 の点を中心とする.
円のグラフィックポインタは中心点です.

【プログラム例】

```
SCREEN 0
CLS 0
CIRCLE (100, 100), 40, , 0, 3.14159
CIRCLE STEP(200, 0), 50
END
```

【結　果】

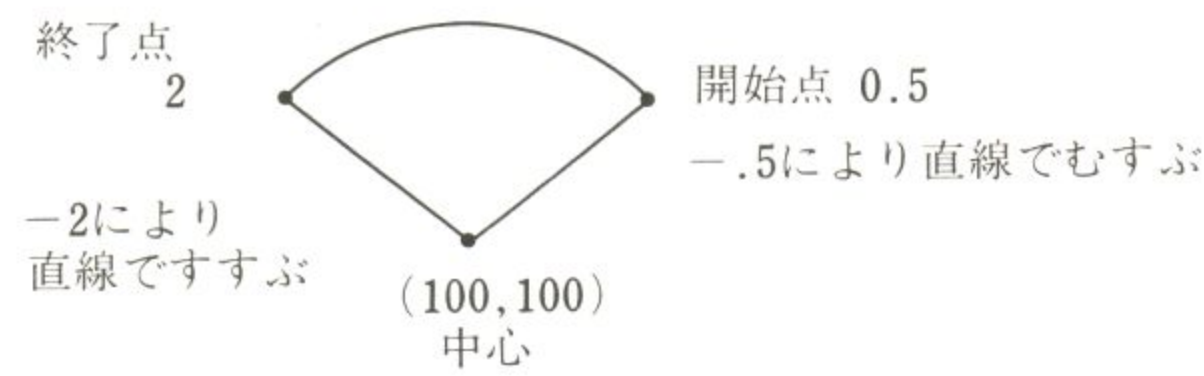
【例題7】　扇形

結果の扇形をかくプログラムをつくれ.

【解　説】　◎　扇形をかきます.

① 開始点と終了点にマイナスをつけると中心と開始点，中心と終了点を直線で結びます．したがって扇形となります．

CIRCLE (100, 100), 40, , -.5, -2

終了点
2
開始点 0.5
－.5により直線でむすぶ
－2により
直線ですすぶ
(100,100)
中心

【プログラム例】

```
SCREEN 0
CLS 0
CIRCLE (100, 100), 40, , -.5, -2
END
```

【結　果】

【例題8】　楕円

結果の楕円を2つかくプログラムをつくれ．

【解　説】　◎楕円をかきます．

① CIRCLE文の最後のオプションの値で円を変形して楕円とします．

CIRCLE STEP(200,0),50,,,,.5

(0,0)からの相対指定　長軸半径　横長の楕円　1未満のとき横長の楕円．1が円

CIRCLE STEP(200,0),50,,,,1.5

(200,0)からの相対指定　長軸半径　縦長の楕円　1をこえるとき縦長の楕円

【プログラム例】

```
SCREEN 0
CLS 0
CIRCLE (200, 100), 50, , , , .5
CIRCLE STEP(200, 0), 50, , , , 1.5
END
```

【結　果】

【例題 9】　ドットの表示

1 ドットを表示することのくり返しで点線をかくプログラムをつくれ.

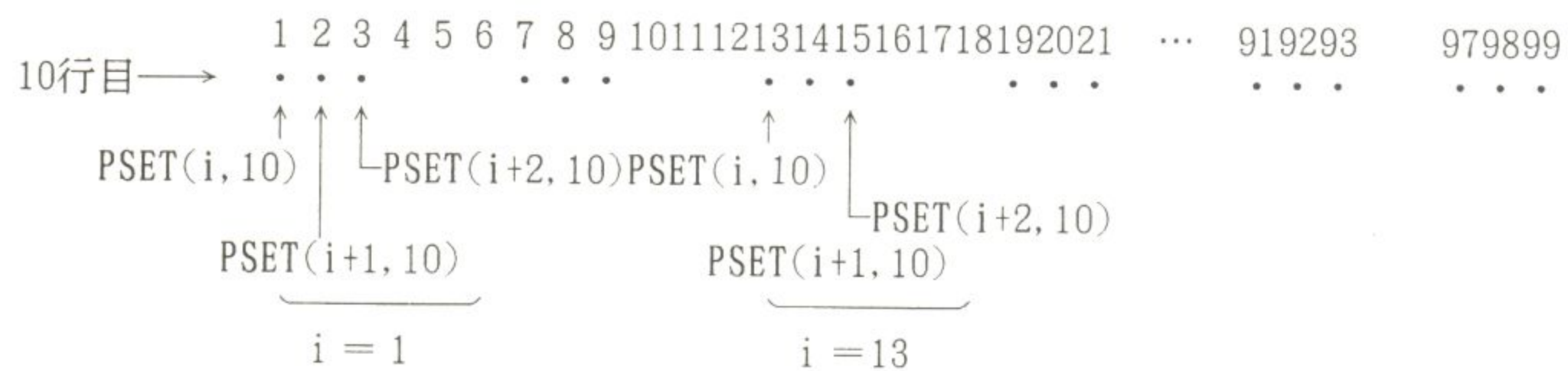
【解　説】　◎　ドットの表示をします.

① PSET は 1 ドットを表示します. 表示位置は(x,y)で指定します.

② 10 ドット行目を指定して, i を 1 から 100 まで増分を 6 でくり返し, i, i+1, i+2 を表示し, i+3, i+4, I+5 を非表示とすると点線となります.

10行目──→ 1 2 3 4 5 6 7 8 9 101112131415161718192021 … 919293 979899

PSET(i, 10)　PSET(i+1, 10)　PSET(i+2, 10)　i = 1

PSET(i, 10)　PSET(i+1, 10)　PSET(i+2, 10)　i =13

【プログラム例】

```
SCREEN 0
CLS 0
FOR i = 1 TO 100 STEP 6
    PSET (i, 10)
    PSET (i + 1, 10)
    PSET (i + 2, 10)
NEXT i
END
```

【結　果】

【例題 10】　ドットの表示を非表示とする

横線を実線でかき, 次にそれを点線にかえるプログラムをつくれ.

【解　説】　◎　表示されているドットを非表示とします.

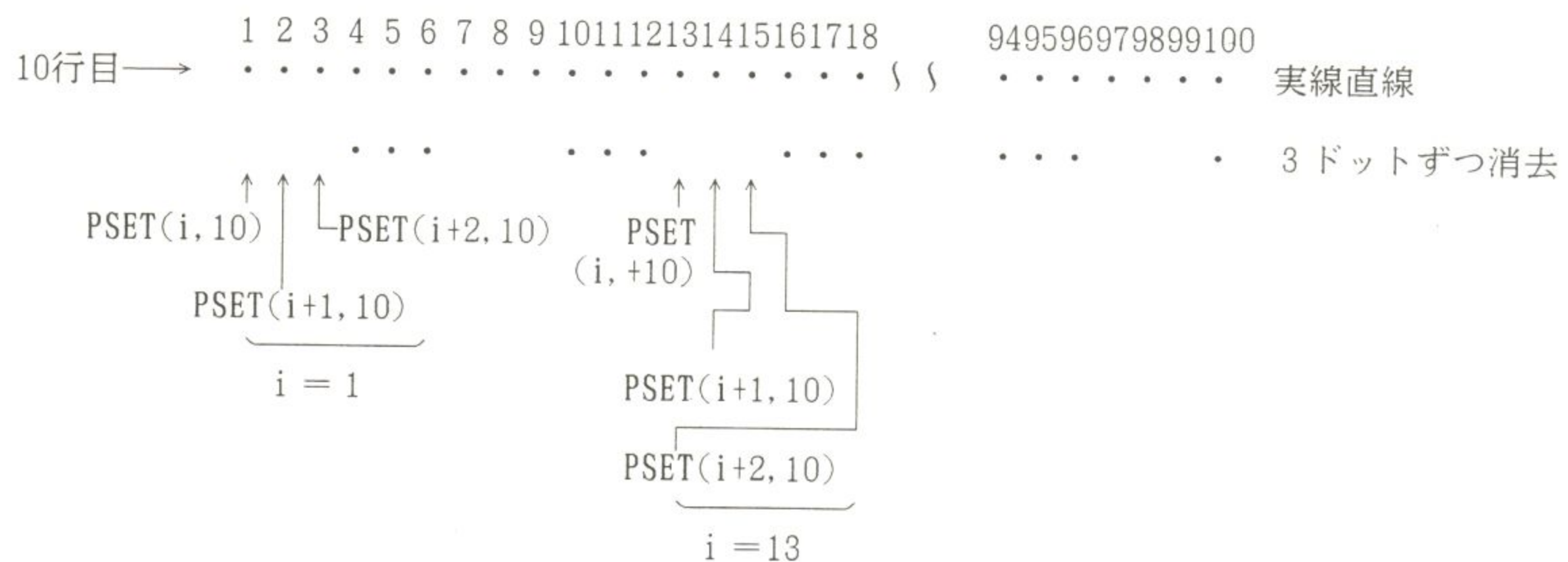
① PRESET(x,y)で(x,y)のドットを非表示とします.

② (1,10)と(100,10)をむすぶ実線をかき, 3 ドットずつ消去すると点線となります.

10行目──→ 1 2 3 4 5 6 7 8 9 101112131415161718 949596979899100 実線直線

3 ドットずつ消去

PSET(i, 10)　PSET(i+1, 10)　PSET(i+2, 10)　i = 1

PSET (i, +10)　PSET(i+1, 10)　PSET(i+2, 10)　i =13

【プログラム例】

```
SCREEN 0
CLS 0
LINE (1, 10)-(100, 10)
FOR i = 1 TO 100 STEP 6
    PRESET (i, 10)
    PRESET (i + 1, 10)
    PRESET (i + 2, 10)
NEXT i
END
```

【結　果】

〔演習問題〕

(1)　ひし形をかくプログラムをつくれ.

(2)　結果の図をつくるプログラムをつくれ.

(3)　点線の円をかくプログラムをつくれ.

(4)　1 点鎖線の円をかくプログラムをつくれ.

(5)　結果の図をつくるプログラムをつくれ.

(6)　結果の図をつくるプログラムをつくれ.

(7)　中心を（300，200）とし，半径を 20，40，60，80，100 の同心円をかくプログラムをつくれ.

(8)　結果の図をかくプログラムをつくれ.

〔解　答〕

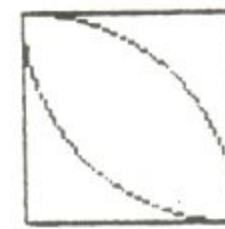
(1)　【プログラム例】

```
SCREEN 0
CLS 0
LINE (100, 100)-(130, 150)
LINE -(100, 200)
LINE -(70, 150)
LINE -(100, 100)
END
```

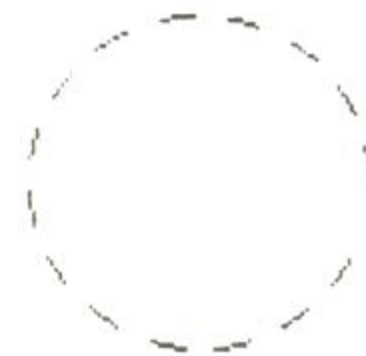
【結　果】

(2)　【プログラム例】

```
SCREEN 0
CLS 0
LINE (100, 100)-(150, 150), , B
CIRCLE (100, 150), 50, , 0, 3.14159 / 2
CIRCLE (150, 100), 50, , 3.14159, 1.5 * 3.14159
END
```

【結　果】

(3)　【プログラム例】

```
SCREEN 0
CLS 0
FOR i = 0 TO 2 * 2.94159 STEP .4
    CIRCLE (100, 150), 40, , i, i + .2
NEXT i
END
```

【結　果】

(4)　【プログラム例】

```
SCREEN 0
CLS 0
FOR i = 0 TO 2 * 2.89159 STEP .5
    CIRCLE (100, 150), 40, , i, i + .2
    CIRCLE (100, 150), 40, , i + .35, i + .35
NEXT i
END
```

【結　果】

(5)　【プログラム例】

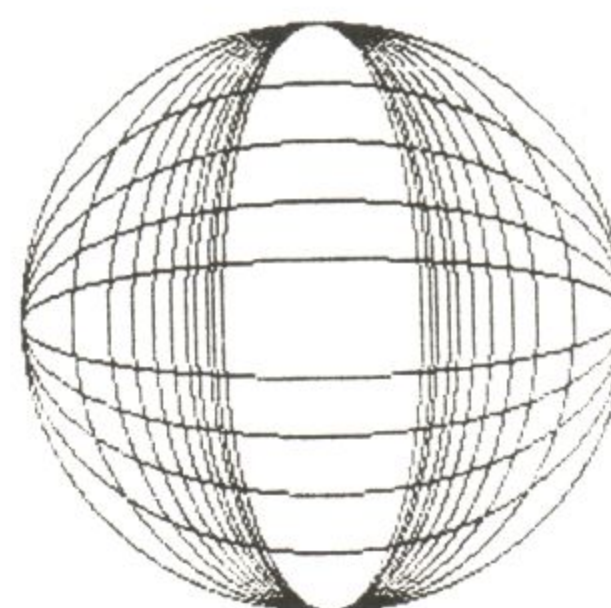
```
SCREEN 0
CLS 0
FOR i = 1 TO 15
    LINE (i * 20, i * 20)-(i * 20 + 10, i * 20 + 10), , B
NEXT i
END
```

【結　果】

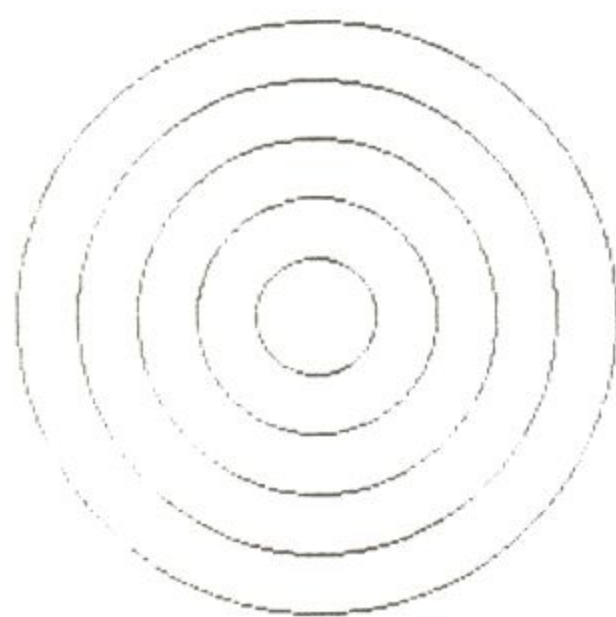
□
□
□
□
□
□
□
□
□
□
□
□
□
□
□

(6)　【プログラム例】

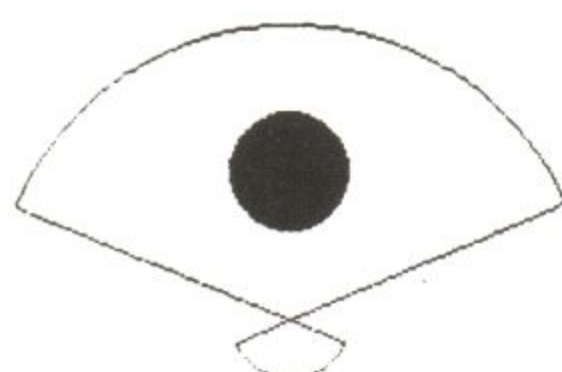
```
SCREEN 0
CLS 0
FOR i = 1 TO 20
    CIRCLE (300, 200), 100, , , , .2 * i
NEXT i
END
```

【結　果】

(7)　【プログラム例】

```
SCREEN 0
CLS 0
FOR i = 20 TO 100 STEP 20 ─┐── 半径を 20, 40, 60, 80, 100 として同心円を表示
    CIRCLE (300, 200), i   │
NEXT i ────────────────────┘
END
```

【結　果】

(8)　【プログラム例】

```
SCREEN 0
CLS 0
CIRCLE (300, 200), 100, , -3.14159 * 1 / 8, -3.14159 * 7 / 8 ──────── 扇形
CIRCLE (300, 200), 20, , -2 * 3.14159 * 9 / 16, -2 * 3.14159 * 15 / 16 ─── 扇形
CIRCLE (300, 150), 20 ─────────────────────────────────────────────── 円
PAINT (300, 150), 7, 7 ────────────────────────────────────────────── 塗りつぶし
END
```

【結　果】

2章　塗りつぶし;タイルパターン

PAINT

【例題 11】　塗りつぶし

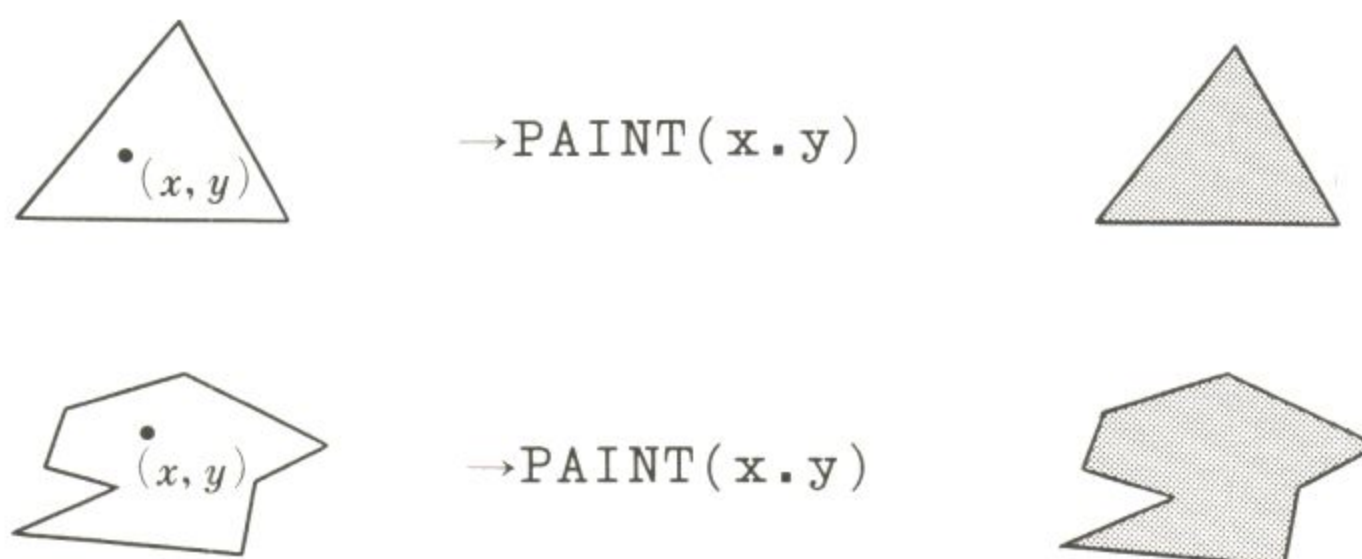
三角形を塗りつぶすプログラムをつくれ.

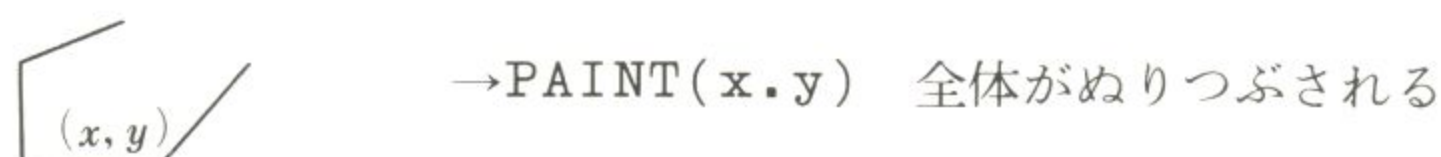
【解　説】　◎　塗りつぶしをします.

① 閉じた境界内を塗りつぶすことができます.

PAINT(x,y) で(x,y) を含む閉じた境界内を塗りつぶします.

(x, y)　→PAINT(x.y)

(x, y)　→PAINT(x.y)

② 閉じていないときは全体を塗りつぶします.

(x, y)　→PAINT(x.y)　全体がぬりつぶされる

【プログラム例】

```
SCREEN 0
CLS 0
LINE (50, 10)-STEP(20, 30)
LINE -STEP(-40, 0)
LINE -STEP(20, -30)
PAINT (50, 20)
END
```

【結　果】

【例題 12】　タイルパターン

結果の5種類のタイルパターンで5つの長方形をうめるプログラムをつくれ.

【解　説】　◎　タイルパターンをつくり，そのパターンで箱を埋めます.

①　タイルパターンは次のようにつくります.

									CHR$(&H0)+CHR$(&H0)+CHR$(&H0)+CHR$(&HFF) 下の4つの8ピクセルの状態の和が1行8ピクセルの状態
	↑							↑	
B	0	0	0	0	0	0	0	0	&H00　(B(青)の8のピクセルの状態)
G	0	0	0	0	0	0	0	0	&H00　(G(緑)の8のピクセルの状態)
R	0	0	0	0	0	0	0	0	&H00　(R(赤)の8のピクセルの状態)
輝度	1	1	1	1	1	1	1	1	&HFF　(輝度の8のピクセルの状態)

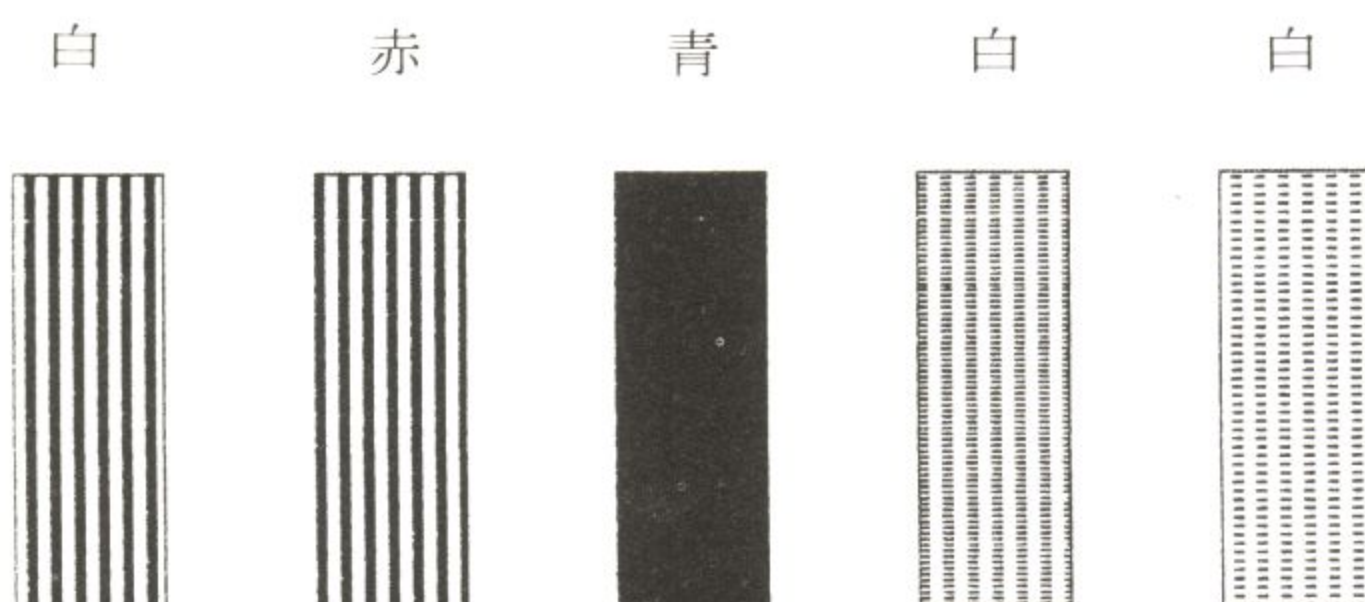
②　2行のパターンをつくるときは1行分のCHR$の和の後に4つのCHR$の和を加えます.

③　3行以上の場合も同様に1行につき4つのCHR$の和をつなぎます.

【プログラム例】

```
SCREEN 0
CLS 0
FOR i = 1 TO 5 ─────────────────────┐─ 5つの四角形を表示
LINE (i * 100, 100)-(i * 100 + 50, 250), , B │
NEXT i ─────────────────────────────┘
t1$ = CHR$(&HF0) + CHR$(&HF0) + CHR$(&HF0) + CHR$(&HFF)─タイルパターン1
PAINT (120, 120), t1$, 7 ───────────────────タイルパターン1で左の箱をうめる
t2$ = CHR$(&H0) + CHR$(&H0) + CHR$(&HF0) + CHR$(&HFF)──タイルパターン2
PAINT (220, 120), t2$, 7 ───────────────────タイルパターン2で左から2番目の箱をうめる
t3$ = CHR$(&HFF) + CHR$(&H0) + CHR$(&H0) + CHR$(&HFF)──タイルパターン3
PAINT (320, 120), t3$, 7 ───────────────────タイルパターン3で真中の箱をうめる
t4$ = CHR$(&H0) + CHR$(&H0) + CHR$(&H0) + CHR$(&H0)───タイルパターン4
t5$ = t1$ + t4$ ────────────────────────タイルパターン5
PAINT (420, 120), t5$, 7───────────────────タイルパターン5で右から2番目の箱をうめる
t6$ = t5$ + t4$ + t4$ + t4$ ─────────────────タイルパターン6
PAINT (520, 120), t6$, 7 ───────────────────タイルパターン6で右側の箱をうめる
END
```

【結　果】

白　　赤　　青　　白　　白

〔演習問題〕

(9)　結果の図をつくるプログラムをつくれ.

(10)　結果の図をつくるプログラムをつくれ.

(11)　結果の５つのパターンで円をぬるプログラムをつくれ.

〔解　答〕

(9)　**【プログラム例】**

```
SCREEN 0
CLS 0
FOR i = 1 TO 10
    CIRCLE (300, 200), i * 10
    IF (i MOD 2) = 0 THEN PAINT (300 + i * 10 - 5, 200), 7, 7
NEXT i
END
```

【結　果】

(10)　**【プログラム例】**

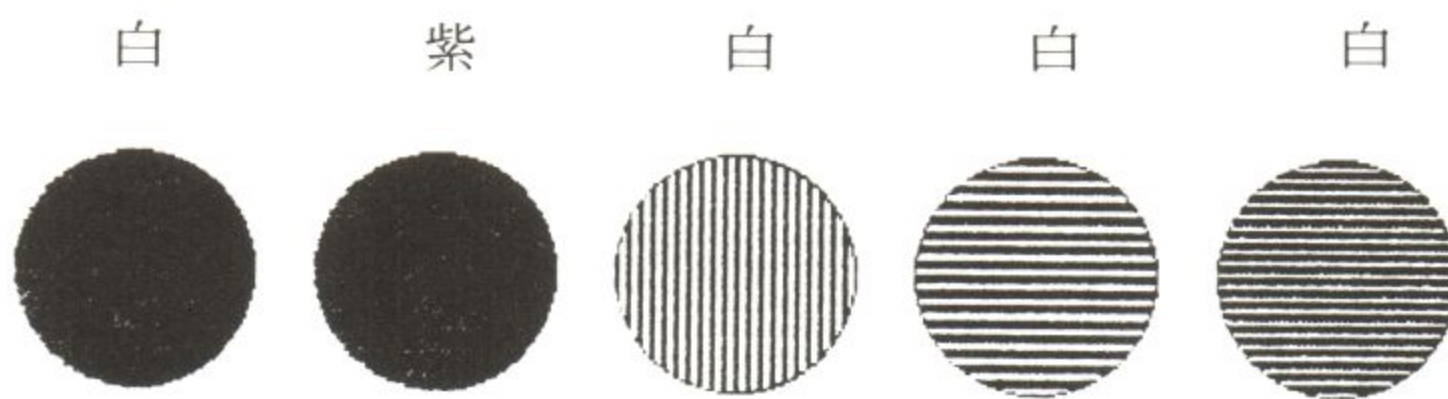
```
SCREEN 0
CLS 0
LINE (100, 100)-(150, 150), , B
CIRCLE (100, 150), 50, , 0, 3.14159 / 2
CIRCLE (150, 100), 50, , 3.14159, 1.5 * 3.14159
PAINT (125, 125), 7, 7
END
```

【結　果】

(11)【プログラム例】

```
SCREEN 0
CLS 0
FOR i = 1 TO 5                                                       ─┐ 5つの円をかく
CIRCLE (i * 100, 100), 40                                             │
NEXT i                                                               ─┘
t1$ = CHR$(&HFF) + CHR$(&HFF) + CHR$(&HFF) + CHR$(&HFF) ──── タイルパターン1
PAINT (100, 100), t1$, 7 ──── タイルパターン1で円をぬる
t2$ = CHR$(&HFF) + CHR$(&H0) + CHR$(&HFF) + CHR$(&HFF) ──── タイルパターン2
PAINT (200, 100), t2$, 7 ──── タイルパターン2で円をぬる
t3$ = CHR$(&HCC) + CHR$(&HCC) + CHR$(&HCC) + CHR$(&HFF) ──── タイルパターン3
PAINT (300, 100), t3$, 7 ──── タイルパターン3で円をぬる
t4$ = CHR$(&H0) + CHR$(&H0) + CHR$(&H0) + CHR$(&H0) ──── タイルパターン4
t5$ = t1$ + t1$ + t4$ + t4$ ──── タイルパターン5
PAINT (400, 100), t5$, 7 ──── タイルパターン5で円をぬる
t6$ = t1$ + t3$ + t4$ ──── タイルパターン6
PAINT (500, 100), t6$, 7 ──── タイルパターン6で円をぬる
END
```

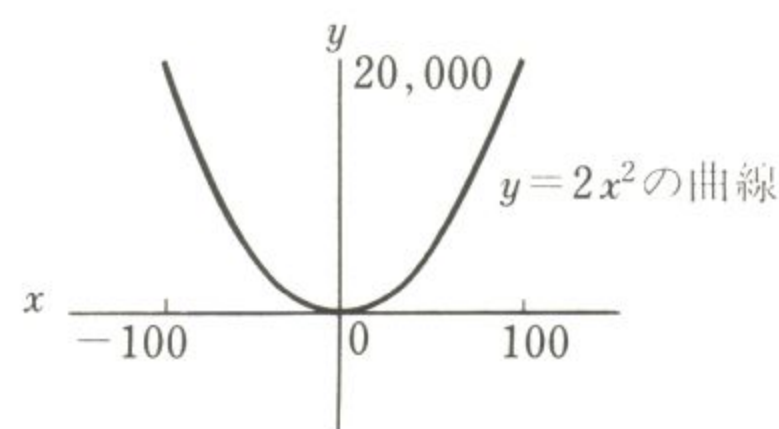
【結　果】

白　　紫　　白　　白　　白

3章　関数式の曲線

【例題 13】　関数式の表示

y=2x² の曲線を x を-100 から+100 までの値としたときの y の値の曲線を表示するプログラムをつくれ.

【解　説】　◎　y=2x² の関数式の曲線を表示します.

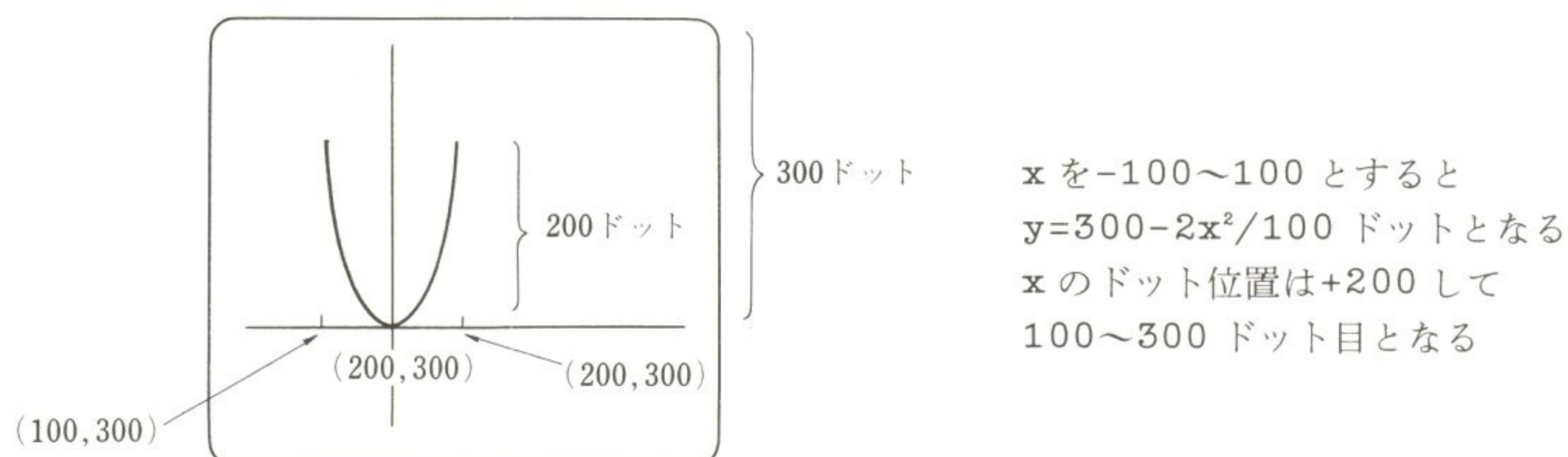
① y=2x² の式で x を-100 から 100 までとすると y の値は最大が 20,000, 最小が 0 の曲線となります.

② この曲線を 640*400 ドットの画面に表示することを考えます.

③ まず x 方向の-100 から 100 は画面上では+200 すなわち, 100 ドットから 300 ドットとします.

④ y 方向の最大 20,000 は 200 ドットに縮小します.

⑤ y 方向の 0 の位置は画面の 300 ドット目とします. なお, 画面は xy 方向は左上が(0,0)で右下が(639,399)ですから y 方向は普通の座標系とは方向が逆です. 画面は y のプラスが下, y のマイナスが上となっています.

⑥ したがって, y=2x² は画面上では次のようになります.

x を-100～100 とすると
y=300-2x²/100 ドットとなる
x のドット位置は+200 して
100～300 ドット目となる

⑦ 実際の表示をします.

y=2x² を fna(x)=2*x*x と定義します.

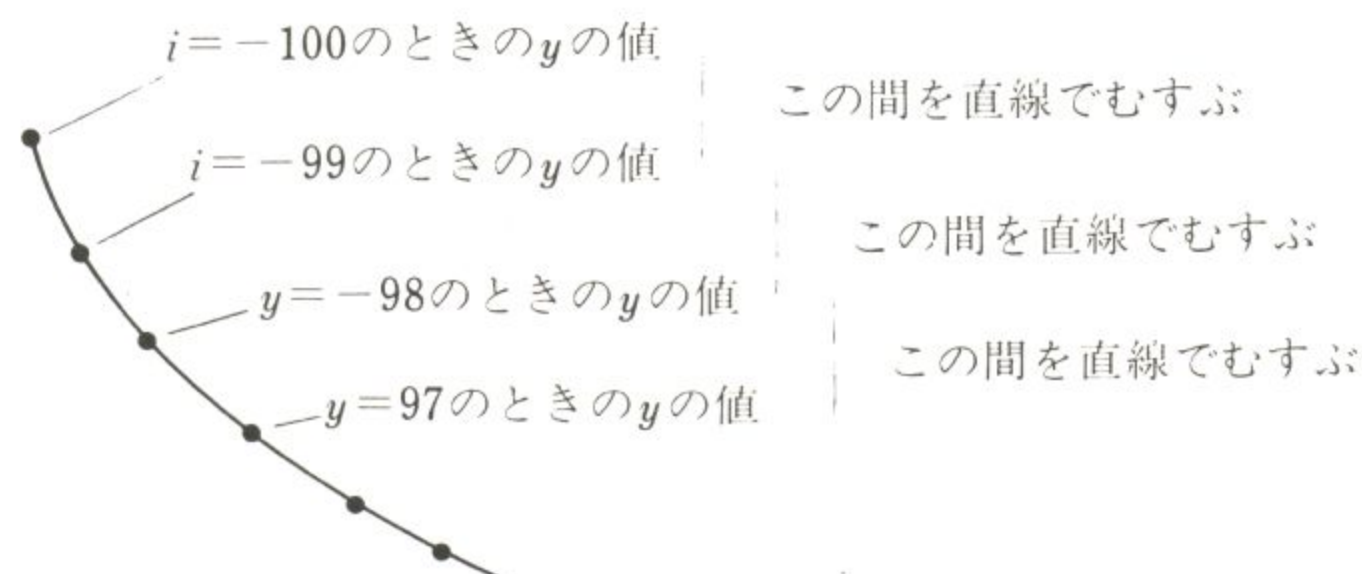
i を-100 から 99 までくり返し

x ドットを i+100　　　としします.

y ドットを 300-fna(i)/100　　これが x が i のときの y の値です.

x ドットを(i+1)+100　　　とします.

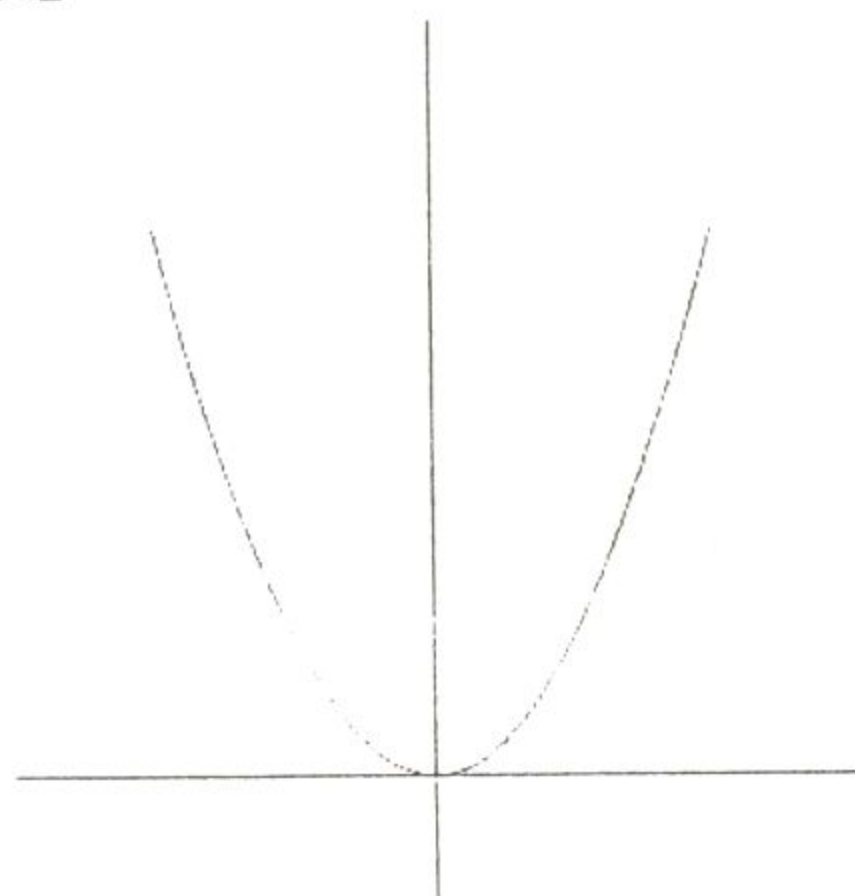
y ドットを 300-fna(i+1)/100 とします. これが x が i+1 のときの y の値です.

この 2 点をむすびます. このくり返しで曲線をかきます.

⑧　表示の仕方としては, x を-100 としたときの y の値の位置にグラフィックポインタを移し, 次に, x を-99, -98, …0, …100 と変化させて y の値を求めその位置を次々に結ぶ方法もあります.

```
PSET(100,300-fna(-100)/100)
FOR i=-100 TO 100
x=i+200
y=300-fna(i)/100
LINE-(x,y)
NEXT i
```

【プログラム例】

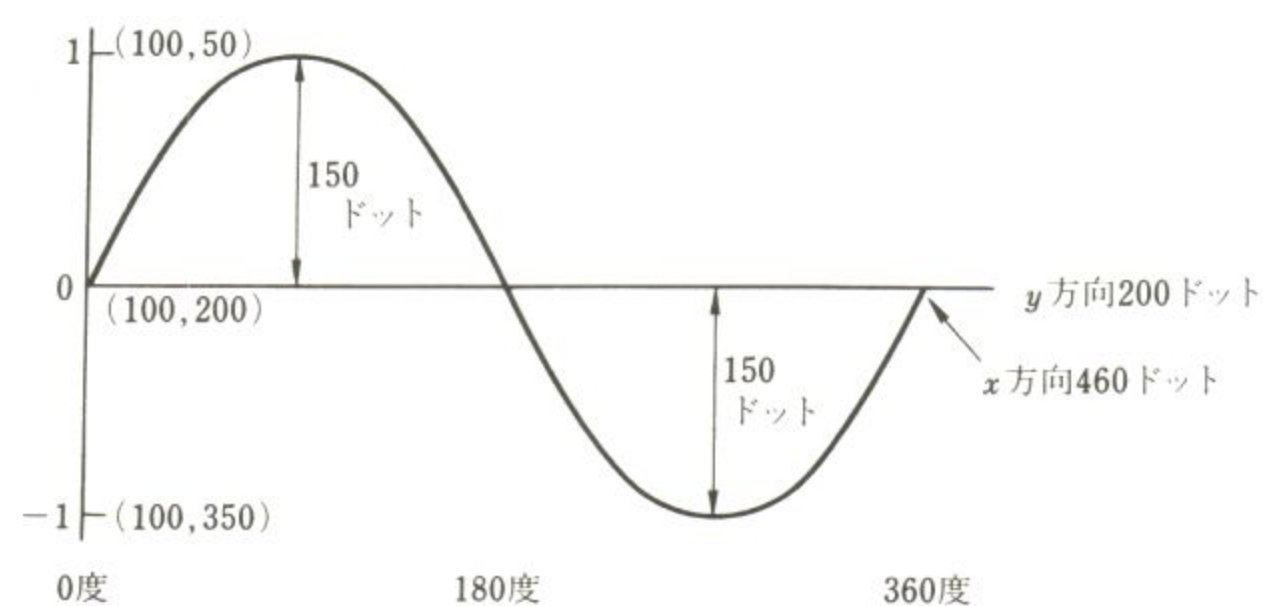
```
SCREEN 0
CLS 0
DEF fna (x) = 2 * x * x
LINE (50, 300)-(350, 300)
LINE (200, 10)-(200, 350)
FOR i = -100 TO 99
    y = 300 - fna(i) / 100
    yy = 300 - fna(i + 1) / 100
    LINE (i + 200, y)-(i + 201, yy)
NEXT i
END
```

【結　果】

【例題 14】　関数式の図

SIN(x)のカーブを表示するプログラムをつくれ.

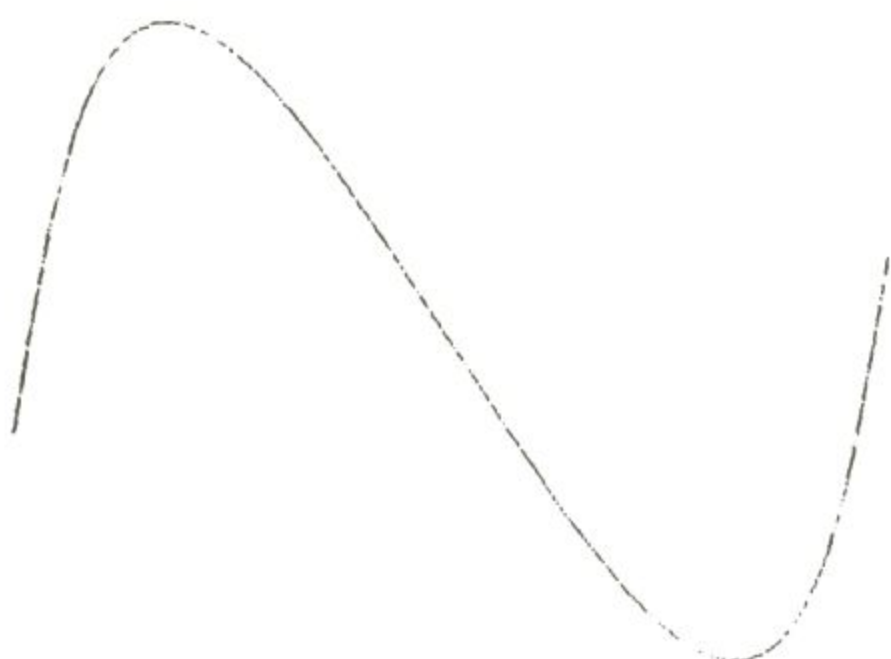
【解　説】　◎　SIN(x)の式の図をかきます.

① SIN(x)は最大値が1，最小値が-1 です．0 のときの y 座標値を 200 として，1 のとき 50，-1 のとき 350 すなわち 1 の幅を 150 ドットとなるようにします．その場合の定義は fny(x)=200-150SIN(x)となります.

x は 0 度から 360 度までとし，0 度のとき x 方向 100 ドット，360 度のとき 460 ドットとします．図は 5 度おきに SIN(x)の値を求めて前の値とつないでいきます.

② SIN(x)で x が 0 度のときの値は(100,fny(0))です．ここにまず点をかき，次は(xx,yy)を求めて順にむすびます．xx は 0 度，5 度，10 度…360 度の x 方向の値で 100，105，110……460 ドットです．yy は sin (0 度)，sin (5 度)，sin (10 度) ……SIN (360 度) のときの y ドットです．fny (x 度) となります.

【プログラム例】

```
SCREEN 0
CLS 0
DEF fny (x) = 200 - 150 * SIN(x)———SIN(x)のy方向のスクリーン上のドット位置
PSET (100, fny(0))—————————————SIN(0)のときのドット位置をオンとする
FOR x = 0 TO 360 STEP 5 ———————┬—SIN(0度), SIN(5度)……SIN(360度)
    xx = x + 100               │  のときのx, y座標値を求め順にむすぶ
    r = x / 180 * 3.14159      │
    yy = fny(r)                │
    LINE -(xx, yy) ————————————┘
NEXT x
END
```

【結　果】

〔演習問題〕

(12)　$y=\sin^9(3x)$のグラフをかくプログラムをつくれ.

(13)　$y=100\sin^3(8x)+50\cos(3x)$のグラフをかくプログラムをつくれ.

〔解　答〕

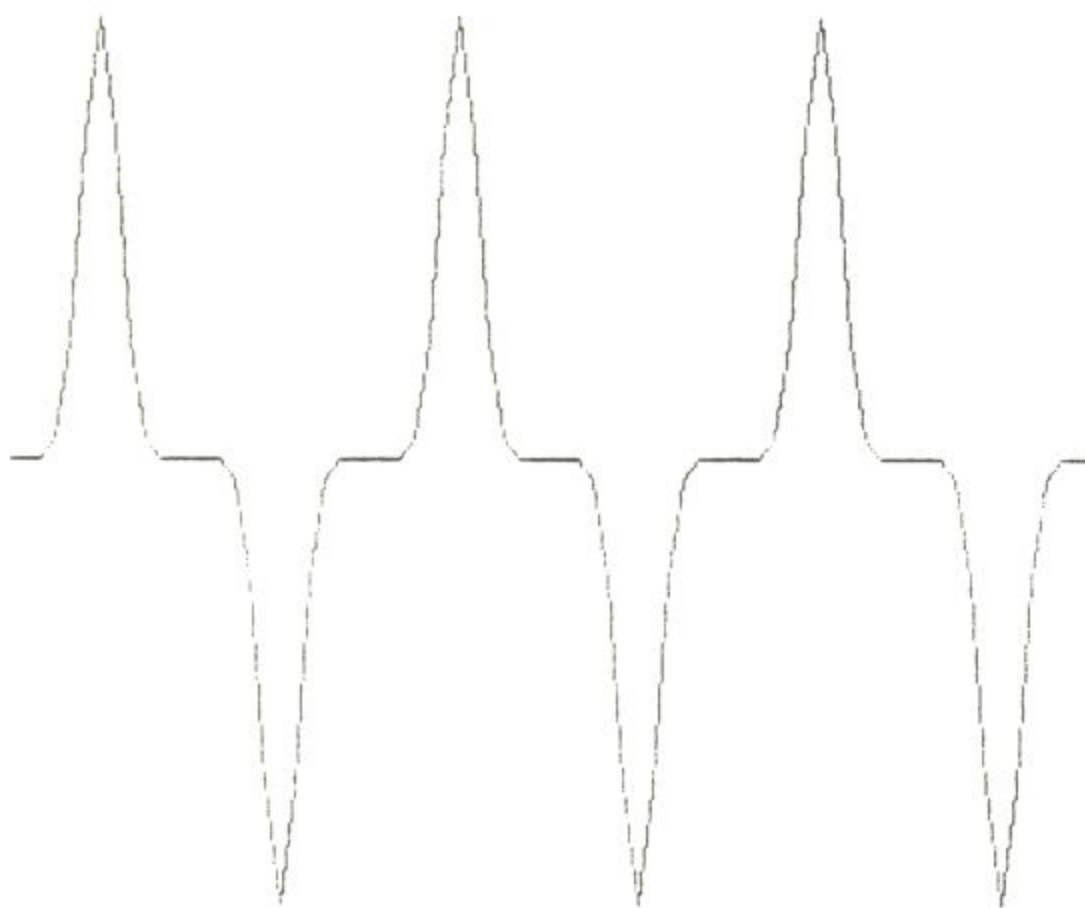
(12)　**【プログラム例】**

```
SCREEN 0
CLS 0
DEF fny (x) = 200 - 150 * SIN(3 * x) ^ 9
PSET (100, fny(0))
FOR x = 0 TO 360 STEP 5
    xx = x + 100
    r = x / 180 * 3.14159
    yy = fny(r)
    LINE -(xx, yy)
NEXT x
END
```

DEF fny …：SIN^9(3x)の値の画面上のｙ座標値を求める式の定義

PSET …：SIN(0)のときのｘ，ｙ座標値のドットをかく

FOR … LINE …：SIN^9(0)～SIN^9(3∗360)の値を５度おきに求め，そのときのｘ，ｙ座標値を順にむすぶ

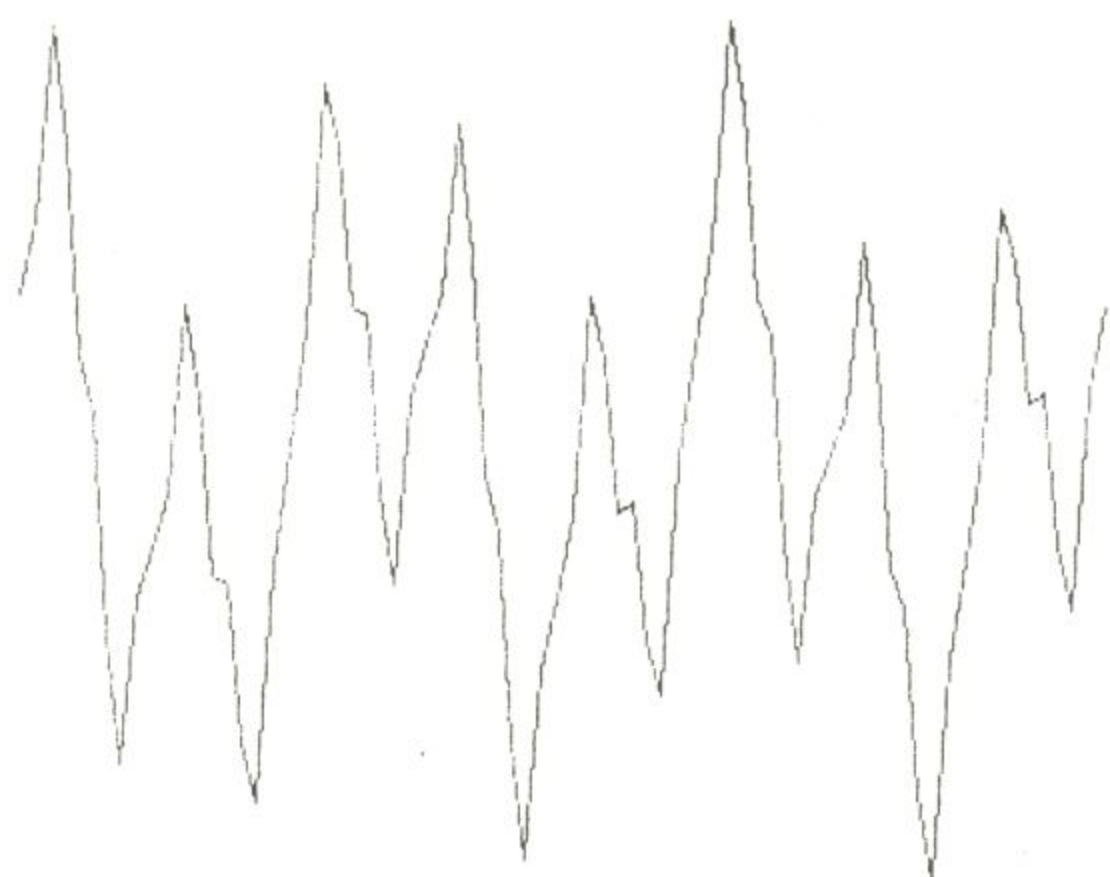
【結　果】

(13)　**【プログラム例】**

```
SCREEN 0
CLS 0
DEF fny (x) = 200 - 100 * SIN(8 * x) ^ 3 - 50 * COS(3 * x)
PSET (100, fny(0))
FOR x = 0 TO 360 STEP 5
    xx = x + 100
    r = x / 180 * 3.14159
    yy = fny(r)
    LINE -(xx, yy)
NEXT x
END
```

DEF fny …：式の定義

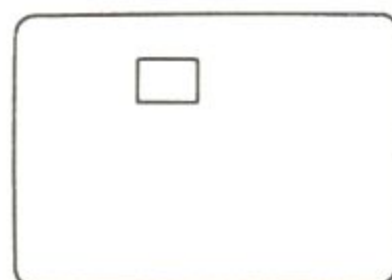
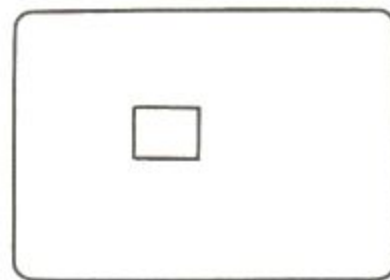
PSET …：０度のときのfnyの値のｘ，ｙ座標値のドットをかく

FOR … LINE …：０度～360度まで５度おきに式の値を求め，ｘ，ｙ座標値を順にむすぶ

【結　果】

4章　画面切りかえウインドウ・ビュー

SCREEN, WINDOW, WINDOW SCREEN, VIEW

【例題 15】　画面の切りかえ

画面 0 に四角形，画面 1 に四角形をかき，画面 0 と 1 を切りかえて 10 回表示するプログラムをつくれ.

画面 0 の四角形　　画面 1 の四角形

【解　説】　◎　画面切りかえ

① Quick BASIC では次のように書き込み画面と表示画面をもっています.

モード	ページ
84	0, 1, 2, 3
87	0 と 1
88, 0	0 と 1

② したがって，SCREEN0 の場合は書き込み，表示おのおの 2 画面ずつもっています．書き込み 2 画面は別々の図をかき，次に表示画面を切りかえると瞬時に図が入れかわります．アニメではよく使う手法です.

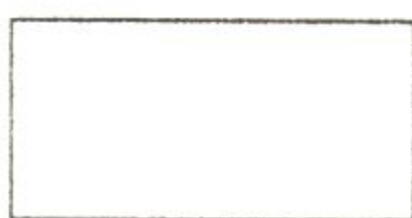
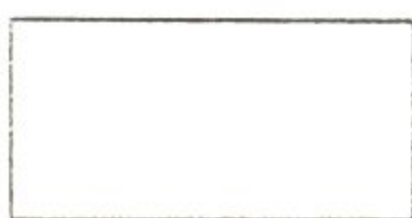
③ 画面 0 と画面 1 の 2 画面をもち，おのおの，独立して，書き込み，表示ができます.

```
        ↓── 0 画面への書き込み
SCREEN 0,,0,0
            ↑
            └──── 0 画面の表示

SCREEN 0,,1,1 ←─ 1 画面の表示
          ↑
          └──── 1 画面への書込み
```

④ 0 画面と 1 画面におのおの図を書き込んでおいて，表示画面を切りかえることで図が瞬時に入れかわります.

【プログラム例】

```
SCREEN 0, , 0, 0
CLS 0
LINE (200, 40)-(300, 90), , B
SCREEN , , 1, 1
LINE (200, 110)-(300, 160), , B
FOR i = 1 TO 10
    SCREEN , , , 0
    FOR j = 1 TO 1000: NEXT j
    SCREEN , , , 1
    FOR j = 1 TO 1000: NEXT j
NEXT i
END
```

【結　果】

（上と下の四角が交互に表示される）

【例題 16】　ウインドウ

(-100,-200)-(200,500)をウインドウとして，(-50,50)と(100,150)を結ぶ直線と，ウインドウを(-500,-1500)-(1000,1000)として(-100,-200)と(200,500)を結ぶ直線をかくプログラムをつくれ．

【解　説】　◎　ウインドウの使い方を示します．

① ワールド座標系の中でかいた図をあるウインドウを設定してその部分を画面へ表示します．これはちょうど，外の景色を窓を通してみるようなものです．窓の大きさ，位置によって表示される図の位置，大きさがかわります．

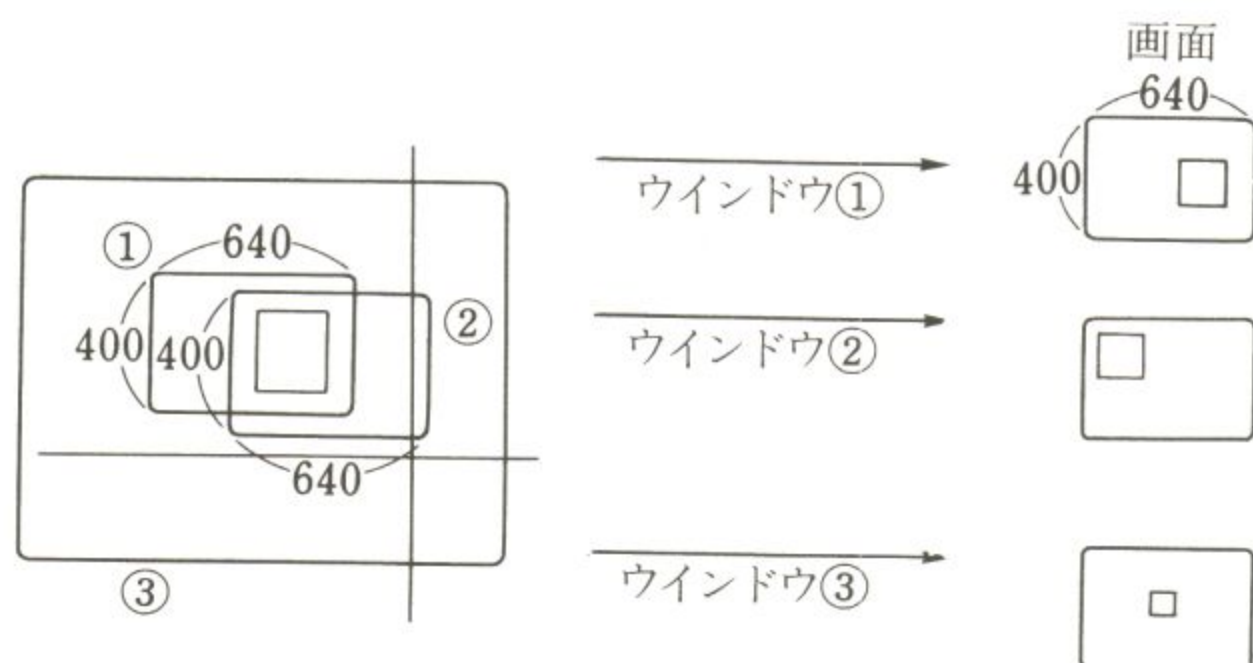
ウインドウの例①

ワールド座標系の中の図（正方形）

画面

640

400

ウインドウ①

ウインドウ②

ウインドウ③

ワールド座標系の中でのウインドウの大きさ＝画面の大きさ

正方形は右側

同上，ただし，ウインドウの位置が異なる

正方形は左上

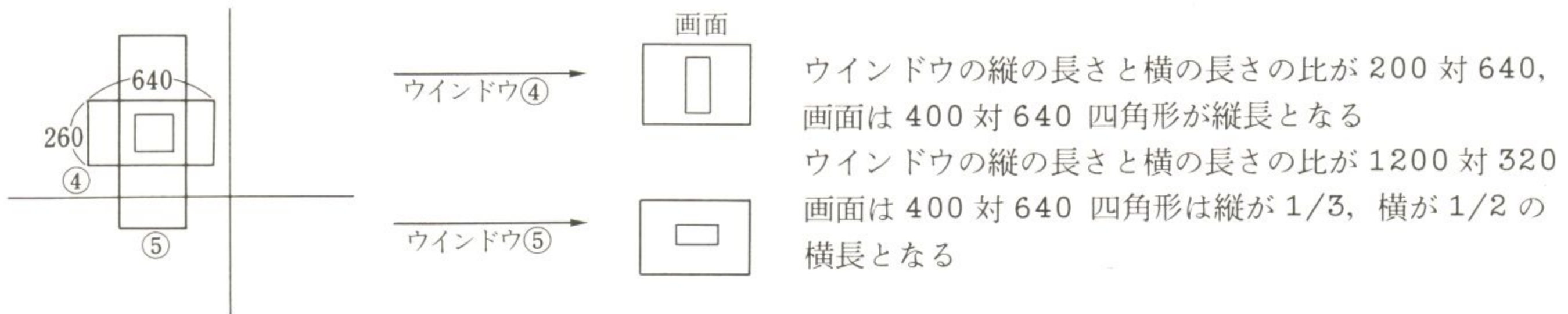
ウインドウの大きさが画面の大きさの 3 倍（縦，横とも）

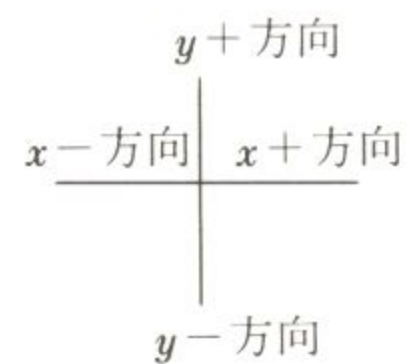
正方形は画面中央で 1/3（縦，横とも）

ウインドウの例②

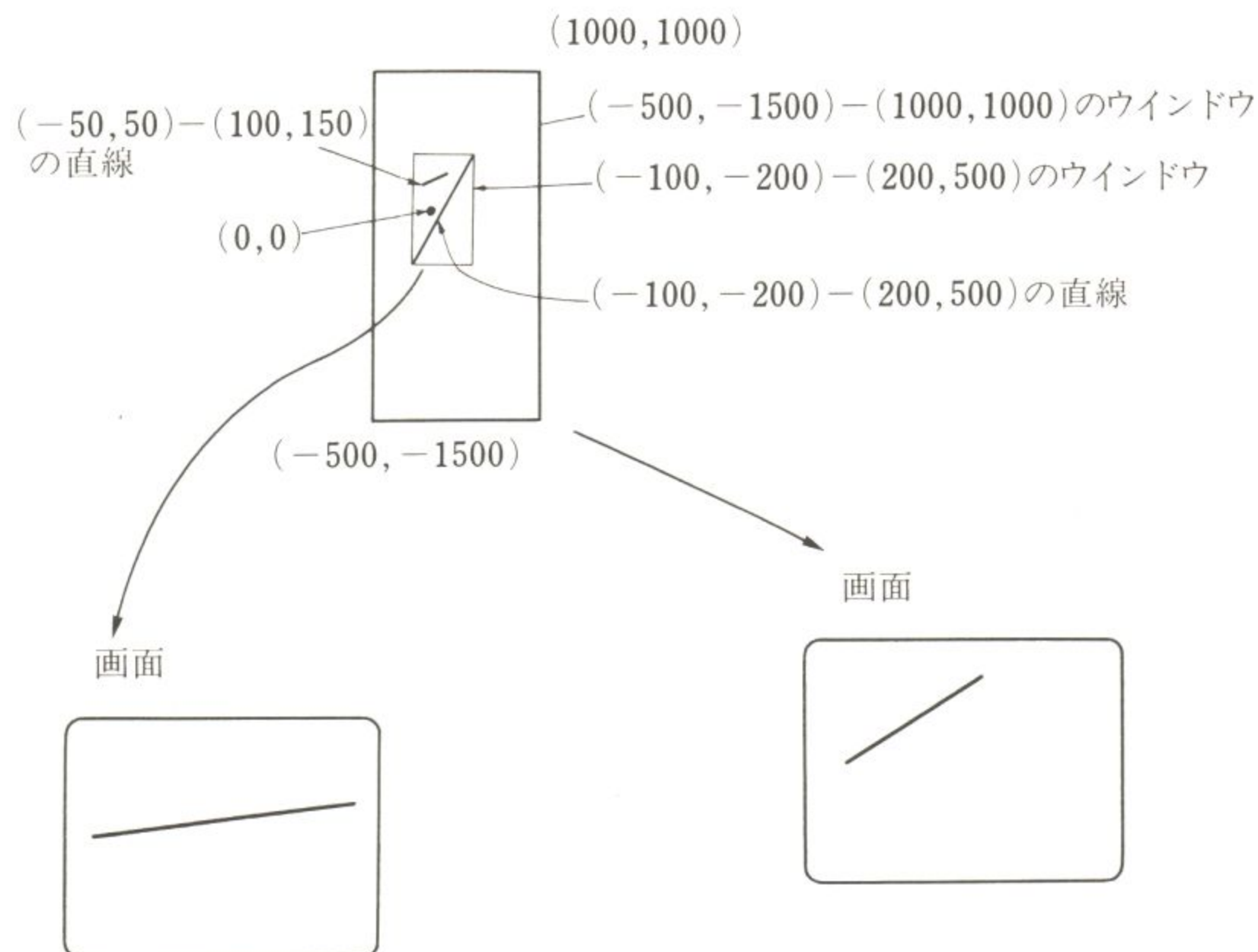
ワールド座標系の中の図（正方形）

640
260
④
⑤
ウインドウ④
ウインドウ⑤
画面

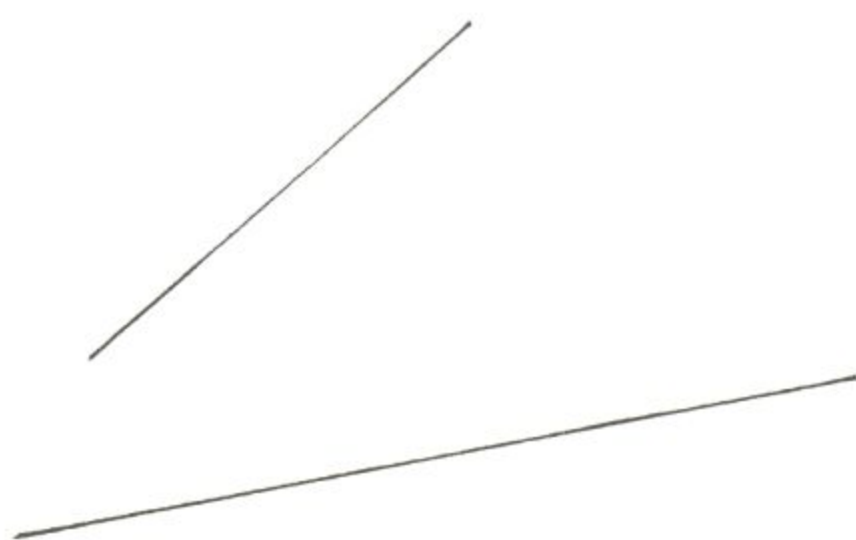
ウインドウの縦の長さと横の長さの比が 200 対 640，画面は 400 対 640 四角形が縦長となる
ウインドウの縦の長さと横の長さの比が 1200 対 320 画面は 400 対 640 四角形は縦が 1/3，横が 1/2 の横長となる

② WINDOW の座標系は次のとおりです.

y＋方向
x－方向　x＋方向
y－方向

③ 項①の図でみたように，切りとったウインドウ全体が画面全体となるように調整されて表示されます.

④ 例題の図は次のようになります.

(1000,1000)
(－500，－1500)－(1000,1000)のウインドウ
(－50,50)－(100,150)の直線
(－100，－200)－(200,500)のウインドウ
(0,0)
(－100，－200)－(200,500)の直線
(－500，－1500)
画面
画面

【プログラム例】

```
SCREEN 0, , 0, 0
CLS 0
WINDOW (-100, -200)-(200, 500)      ウインドウの設定と直線
LINE (-50, 50)-(100, 150)
WINDOW (-500, -1500)-(1000, 1000)   ウインドウの設定と直線
LINE (-100, -200)-(200, 500)
END
```

【結　果】

【例題17】　ウインドウスクリーン

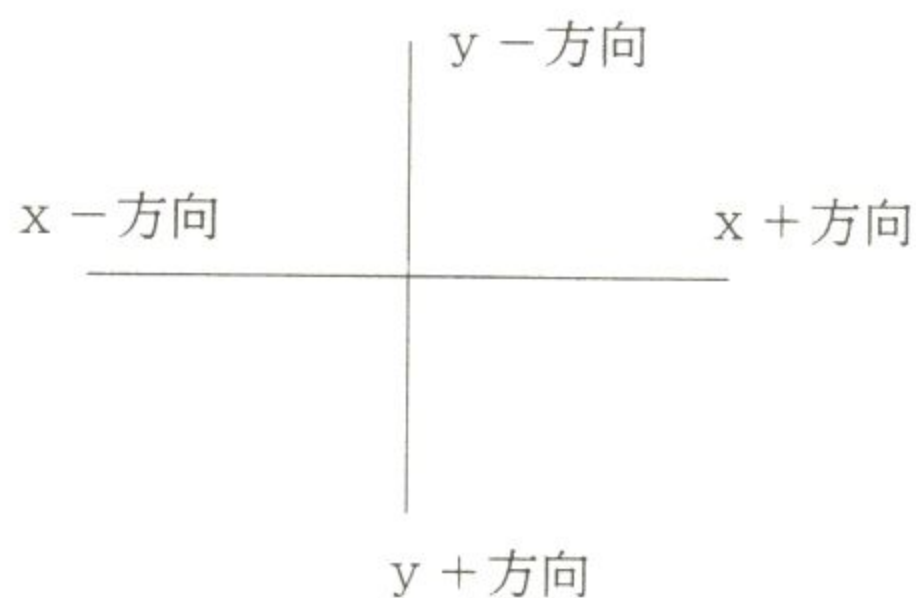
【例題16】の図を逆にして表示するプログラムをつくれ.

【解　説】　◎　ウインドウスクリーンを設定して図を表示します.

①　WINDOW SCREEN は次のような座標系を考えます. WINDOW と y 方向のプラスとマイナスが逆になるほかは全く同じです.

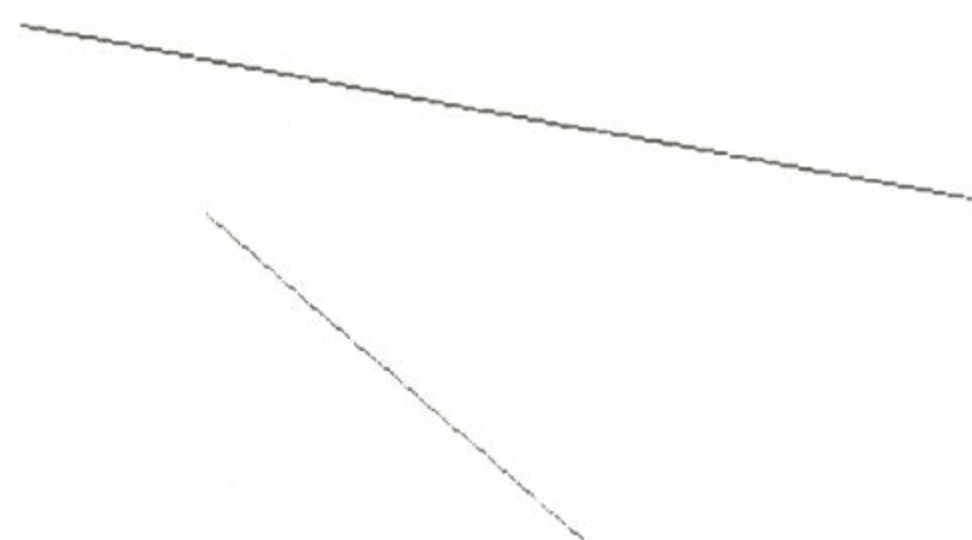
y－方向

x－方向　　x＋方向

y＋方向

②　したがって, WINDOW と WINDOW SCREEN を使い同じ値で設定して同じ図をかくと縦方向が逆となります.

【プログラム例】

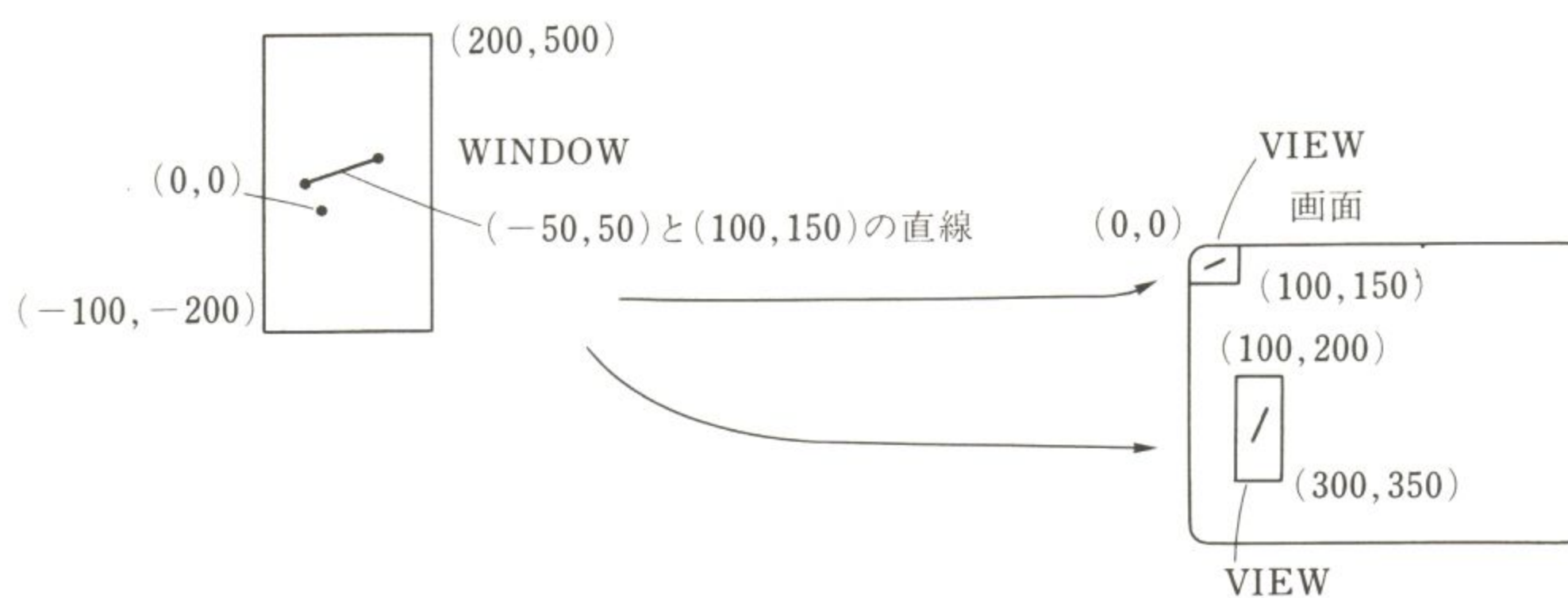
```
SCREEN 0, , 0, 0
CLS 0
WINDOW SCREEN (-100, -200)-(200, 500)──────┐ウインドウの設定と図
LINE (-50, 50)-(100, 150) ─────────────────┘
WINDOW SCREEN (-500, -1500)-(1000, 1000)───┐ウインドウの設定と図
LINE (-100, -200)-(200, 500) ──────────────┘
END
```

【結　果】

【例題18】　ビュー

(-50,50)と(100,150)を結ぶ直線を(-100,-200)と(200,500)を対角とするウインドウで切って(0,0)と(100,50)を対角とするビューに表示し，次に同じ直線を(100,200)と(300,350)を対角とするビューに表示するプログラムをつくれ.

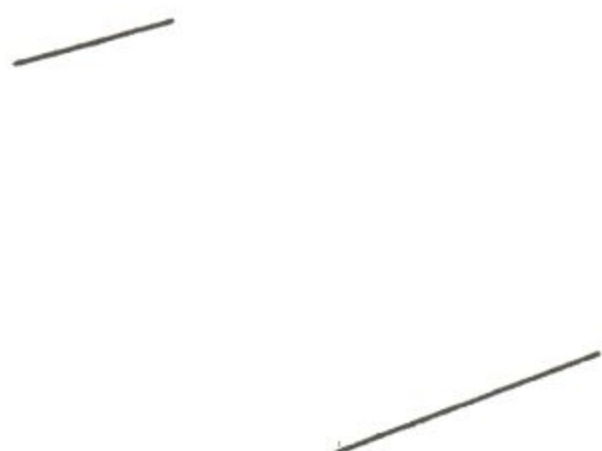
【解　説】　◎　ビューを指定します.

① VIEWは画面上の対角を示して，その範囲にウインドウの図を表示します．したがって，ウインドウで切りとった図を画面の任意の位置に表示できます.

② WINDOW, VIEW, LINEは次のようになります.

③ VIEWの設定のない場合は画面全体がビューです.

【プログラム例】

```
SCREEN 0, , 0, 0
CLS 0
WINDOW (-100, -200)-(200, 500)———ウインドウの設定
VIEW (0, 0)-(100, 50)————————ビューの設定
LINE (-50, 50)-(100, 150)——————直線
VIEW (100, 200)-(300, 350)—————ビューの設定
LINE (-50, 50)-(100, 150)——————直線
END
```

【結　果】

〔演習問題〕

(14)　画面0と画面1に，左矢印と右矢印をかき，画面の切りかえで交互に10回表示するプログラムをつくれ.

(15)　画面０に四角形，画面１にその四角形と同じ四角形の中に円をかいた図をつくり，交互に10回表示するプログラムをつくれ.

(16)　(-1000,-1000)-(1000,1000)をウインドウとして(-500,500)と(100,150)を対角とする箱，ウインドウを(-500,-500)-(1000,1000)として，(-500,500)を(100,150)を対角とする箱，ウインドウを(-1000,-1000)-(500,5000)として(-500,500)と(100,150)を対角とする箱をかくプログラムをつくれ.

(17)　(-100,-200)-(200,500)をウインドウとし，(-50,50)と(100,150)を対角とする箱をビューを(0,0)-(100,100)，(100,100)-(200,200)(0,0)-(639,399)の３種にして表示するプログラムをつくれ.

〔解　答〕

(14) **【プログラム例】**

```
SCREEN 0, , 0, 0 ──────────┐ 左矢印を画面０にかいて表示
CLS 0                      │
LINE (200, 40)-(300, 40)   │
LINE (220, 35)-(200, 40)   │
LINE -(220, 45)────────────┘
SCREEN , , 1, 1 ───────────┐ 右矢印を画面１にかいて表示
LINE (200, 40)-(300, 40)   │
LINE (280, 35)-(300, 40)   │
LINE -(280, 45) ───────────┘
FOR i = 1 TO 10 ───────────┐ 画面０と１を交互に10回表示
    SCREEN , , , 0         │
    FOR j = 1 TO 1000: NEXT j
    SCREEN , , , 1         │
    FOR j = 1 TO 1000: NEXT j
NEXT i ────────────────────┘
END
```

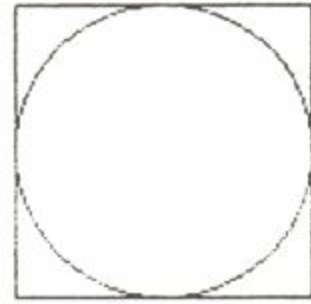
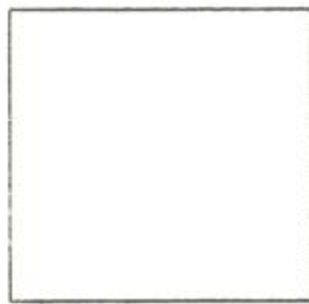
【結　果】

（矢印が同じ場所で交互に表示）

(15) **【プログラム例】**

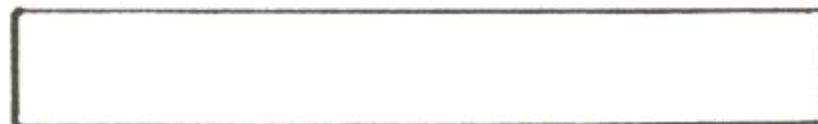
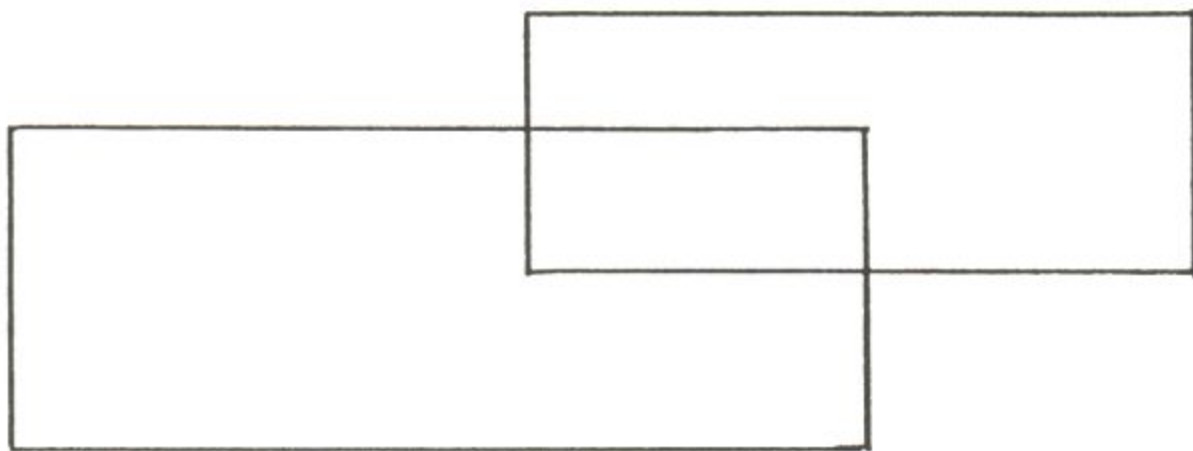
```
SCREEN 0, , 0, 0 ──────────┐ 画面０に四角形をかいて表示
CLS 0                      │
LINE (100, 50)-(200, 150), , B─┘
```

```
SCREEN , , 1, 1 ─────────┐─画面1に四角形と円をかいて表示
LINE (100, 50)-(200, 150), , B   │
CIRCLE (150, 100), 50 ───────┘
FOR i = 1 TO 10 ─────────┐─交互に10回表示
    SCREEN , , , 0               │
    FOR j = 1 TO 1000: NEXT j    │
    SCREEN , , , 1               │
    FOR j = 1 TO 1000: NEXT j    │
NEXT i ──────────────────┘
END
```

【結　果】

左図と右図が同じ場所で交互に表示される

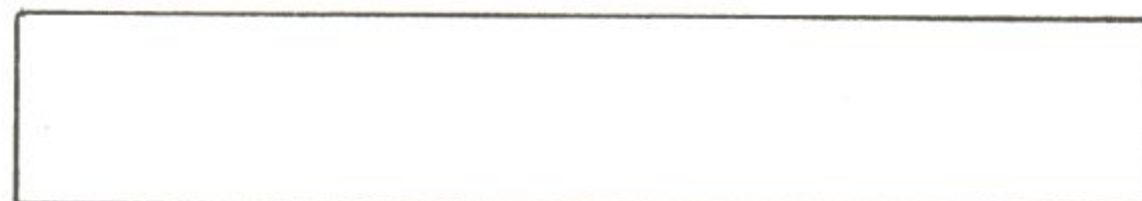
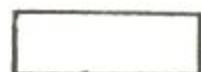
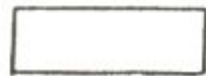
(16)　【プログラム例】

```
SCREEN 0, , 0, 0
CLS 0
WINDOW (-1000, -1000)-(1000, 1000)─┐─ウインドウの設定と箱の表示
LINE (-500, 500)-(100, 150), , B ──┘
WINDOW (-500, -500)-(1000, 1000)───┐─ウインドウの設定と箱の表示
LINE (-500, 500)-(100, 150), , B ──┘
WINDOW (-1000, -1000)-(500, 5000)──┐─ウインドウの設定と箱の表示
LINE (-500, 500)-(100, 150), , B ──┘
END
```

【結　果】

(17)　【プログラム例】

```
SCREEN 0, , 0, 0
CLS 0
WINDOW (-100, -200)-(200, 500) ───ウインドウの設定
VIEW (0, 0)-(100, 100) ───────────ビューの設定
LINE (-50, 50)-(100, 150), , B────四角形の表示
VIEW (100, 100)-(200, 200)────────ビューの設定
LINE (-50, 50)-(100, 150), , B ───四角形の表示
```

```
VIEW (0, 0)-(639, 399) ──────── ビューの設定
LINE (-50, 50)-(100, 150), , B ─── 四角形の表示
END
```

【結　果】

5章　カラー

PALETTE, COLOR, SCREEN

【例題19】　カラー

四角形をかき，次に黒から黄まで6色でかきかえるプログラムをつくれ.

【解　説】　◎　カラーを変化させます.

① 4行目で四角形をかきます. このとき，線の色は4によって　色でかきます. これは標準の指定で3行目のPALETTEを指定します.

0	黒	1	青	2	緑	3	水色
4	赤	5	紫	6	黄	7	白

② 6行目のPALETTE　4, iによって，iが0から6まで変化しますので，4が0の色，1の色……6の色になります. したがって，四角形が黒，青，緑，……となります. なお，黒は背景が黒で色が黒ですから実際はみえません.

PALETTE x,yは値xを色番号yでわりあてます.

③ したがって，同じ赤でかく場合でも例えば次のようにできます.

```
PALETTE 4,4
LINE(100,100)-(200,150),4,B

PALETTE 2,4
LINE(100,100)-(200,150),2,B

PALETTE 5,4
LINE(100,100)-(200,150),5,B
```

【プログラム例】

```
SCREEN 0
CLS 0
PALETTE ───────────────────── 色コードを既定値にもどす
LINE (100, 100)-(200, 150), 4, B ──── 4の色（赤）で四角形をかく
FOR i = 0 TO 6 ──────────┐──── 4に相当する色を0～6の色に変える
    PALETTE 4, i               │
    FOR j = 1 TO 1000: NEXT j  │
NEXT i ──────────────────┘
PALETTE ───────────────────── 色コードを既定値にもどす
END
```

【結　果】

赤，黒，青，緑，水色，赤，紫，黄，赤で表示

【例題 20】　文字の色

"Quick　Basic"を青，赤，水色でかくプログラムをつくれ.

【解　説】　◎　文字の色を指定します.

① a$を"Quick Basic"とします.

② COLOR　1 として PRINT a$を行うと 1 の色番号の色で a$を表示します．すなわち青です．色番号は次の色を示します.

0　黒　　1　青　　2　緑　　3　水色　　4　赤　　5　紫　　6　黄　　7　白

8～15 は 0～7 の反転，16～31 は 0～15 の点滅

【プログラム例】

```
SCREEN 0
CLS 0
a$ = "Quick Basic"
COLOR 1
PRINT a$
COLOR 4
PRINT a$
COLOR 3
PRINT a$
COLOR 7
END
```

【結　果】

```
Quick Basic  (青)
Quick Basic  (赤)
Quick Basic  (水色)
```

〔演習問題〕

(18)　結果の図をつくるプログラムをつくれ.

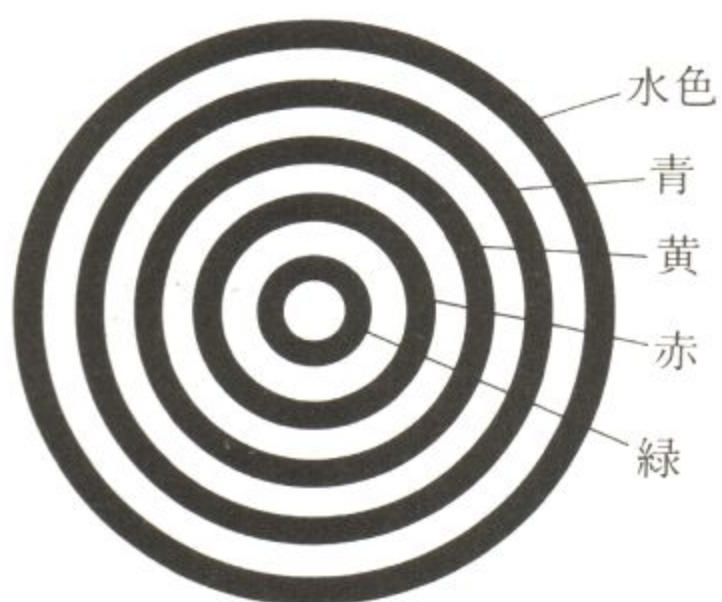
(19)　四角形を 2 つ上下に赤色と緑色で塗り，次に上の四角を赤と青と，下の四角を緑と黄に切りかえることを 10 回くり返すプログラムをつくれ.

〔解　答〕

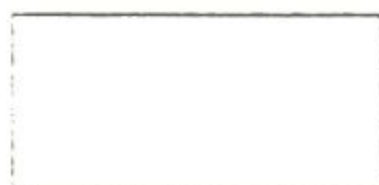
(18)　**【プログラム例】**

```
SCREEN 0
CLS 0
FOR i = 1 TO 10
    CIRCLE (300, 200), i * 10
    IF (i MOD 2) = 0 THEN PAINT (300 + i * 10 - 5, 200), i MOD 7, 7
NEXT i
END
```

【結　果】

水色
青
黄
赤
緑

(19)　**【プログラム例】**

```
SCREEN 0
CLS 0
PALETTE ――――――――――――――― 色コードを既定値にもどす
LINE (100, 100)-(200, 150), 4, BF ―――― 四角を赤で塗る
LINE (100, 200)-(200, 250), 2, BF ―――― 四角を緑で塗る
FOR i = 1 TO 10 ――――――――――― 2つの四角形の色を切りかえる
    PALETTE 4, 4
    PALETTE 2, 2
    FOR j = 1 TO 200: NEXT j
    PALETTE 4, 1
    PALETTE 2, 6
    FOR j = 1 TO 200: NEXT j
NEXT i
PALETTE ――――――――――――――― 色コードで既定値にもどす
END
```

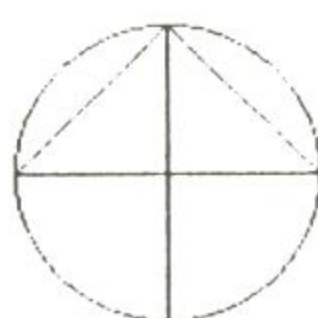
【結　果】

赤と青の切りかえ

緑と黄の切りかえ

6章　図形の配列へのとりこみと表示

GET, PUT

【例題 21】　図形の配列へのとり込みと表示

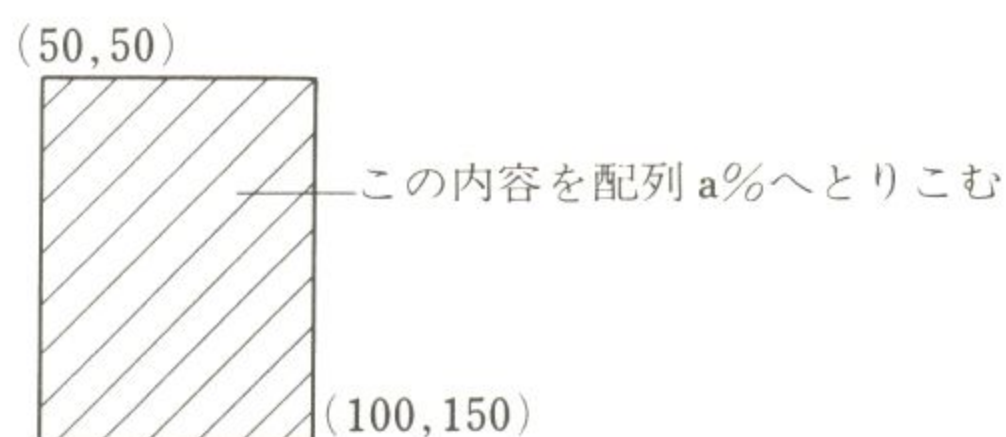
下図を表示して全体を配列 a%，右上 1/4 を配列 b%にとり込んで，次に全体を，(200，50)を左上として，1/4 の図を(350,50)を左上として表示するプログラムをつくれ．

【解　説】　◎　図を配列へとり込み，任意の位置へ表示します．

① 4 行目の CIRCLE 文から 8 行目の LINE 文までで，問題の図をつくります．

② 次の形で図を配列へとり込みます．

```
GET(50,50)-(150,150),a%
```

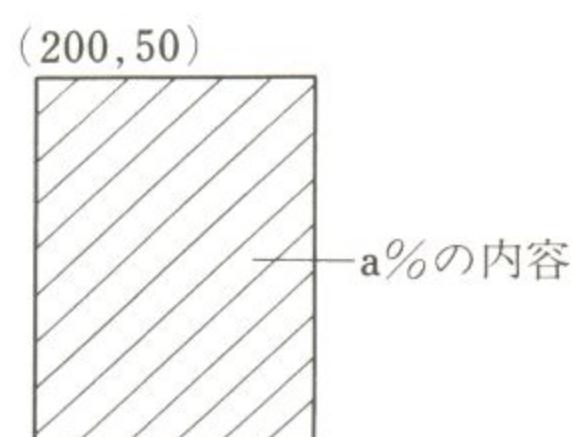
左上と右下を示す座標値を対角とする四角形の内容配列にとりこむ内容

とり込む配列名　配列は前もって充分量を確保しておく

③ 次の形で任意の位置に表示します．

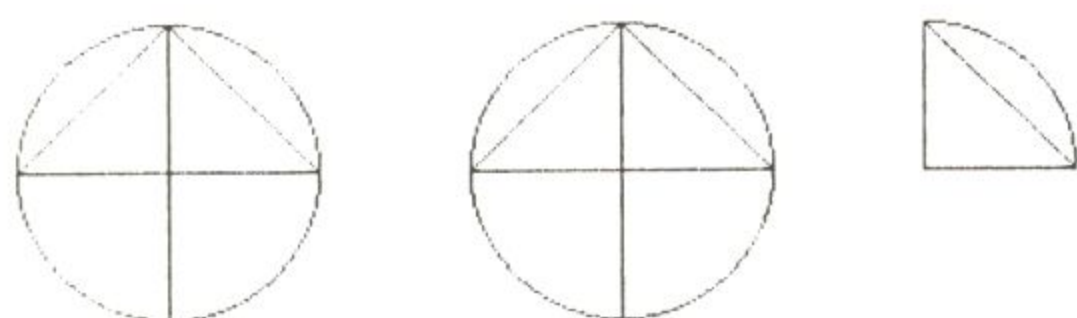
```
PUT(200,50),a%,PSET
```

表示をする左上の座標

表示する配列

表示を示す

【プログラム例】

```
SCREEN 0
CLS 0
DIM a%(2700), b%(800)────────────図をとりこむ配列
CIRCLE (100, 100), 50 ───────┬──図の表示
LINE (100, 50)-(150, 100)        │
LINE -(50, 100)                  │
LINE -(100, 50)                  │
LINE -(100, 150)─────────────────┘
GET (50, 50)-(150, 150), a%──────┬──図形のとり込み
GET (100, 50)-STEP(50, 50), b%───┘
PUT (200, 50), a%, PSET──────────┬──図形を表示
PUT (350, 50), b%, PSET──────────┘
END
```

【結　果】

【例題 22】　オプション付きの表示

【例題 21】と同じ図形をかき全体を配列 a%，右上 1/4 を配列 b%にとりこんで，次に(90,190)-(160,260)と(190,190)-(260,260)の四角形を白でうめておいてから，①a%の内容を左上を(200,50)として PRESET 付き，②b%の内容を左上を(50,50)として OR 付き，③b%の内容を左上を(100,200)として XOR 付き，④b%の内容を左上を(200,200)として AND 付きで表示するプログラムをつくれ.

【解　説】　◎　PUT にオプションをつけて表示します.

① PUT は PSET の他，PRESET，OR，XOR，AND のオプションがつけられます.

② 10 行目までは【例題 21】と同じで図を配列 a%と b%にとり込みます.

③ 11～12 行目は四角形を 2 つ塗りつぶします.

④ PUT で図を表示する場合，PRESET を指定すると，すでに表示がしてある場合は PUT する内容と同じドットは消去，表示のない場合は無表示です.

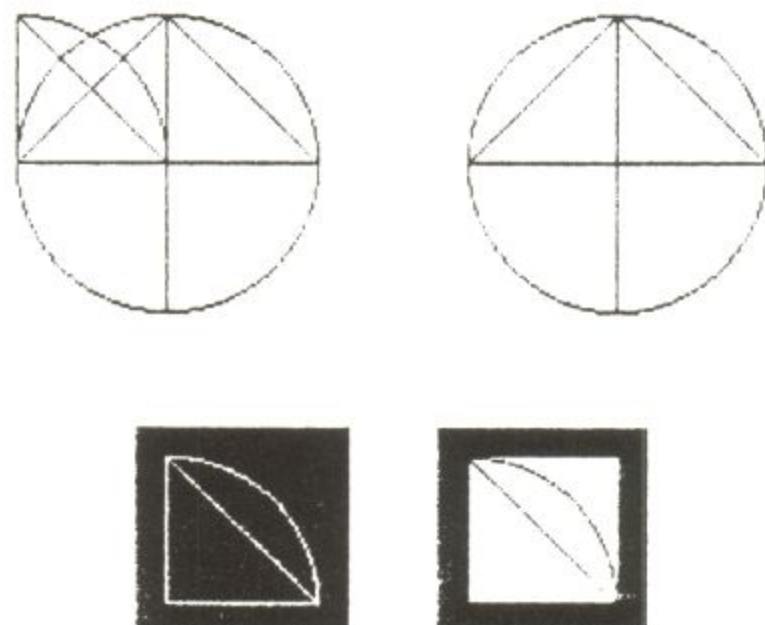
⑤ PUT で表示する場合 OR をつけると，表示するドットと，現在の画面のドットの状態の OR をとってドットの状態を決めます.

⑥ PUT で表示する場合 XOR をつけると表示するドットと，現在の画面のドットの状態の XOR をとってドットの状態を決めます.

⑦ PUT で表示する場合 AND をつけると表示するドットと，現在の画面のドットの状態の AND をとってドットの状態を決めます.

【プログラム例】

```
SCREEN 0
CLS 0
DIM a%(2700), b%(800)
CIRCLE (100, 100), 50
LINE (100, 50)-(150, 100)
LINE -(50, 100)
LINE -(100, 50)
LINE -(100, 150)
GET (50, 50)-(150, 150), a%
GET (100, 50)-STEP(50, 50), b%
LINE (90, 190)-(160, 260), , BF
LINE (190, 190)-(260, 260), , BF
PUT (200, 50), a%, PRESET
PUT (50, 50), b%, OR
PUT (100, 200), b%, XOR
PUT (200, 200), b%, AND
END
```

【結　果】

〔演習問題〕

(20)　結果図の下側の背景の中で塗りつぶした円を左から右へ動かすプログラムをつくれ.

(21)　結果の図をつくり，配列a%にとりこんで，次に画面左から右へ3つ表示するプログラムをつくれ.

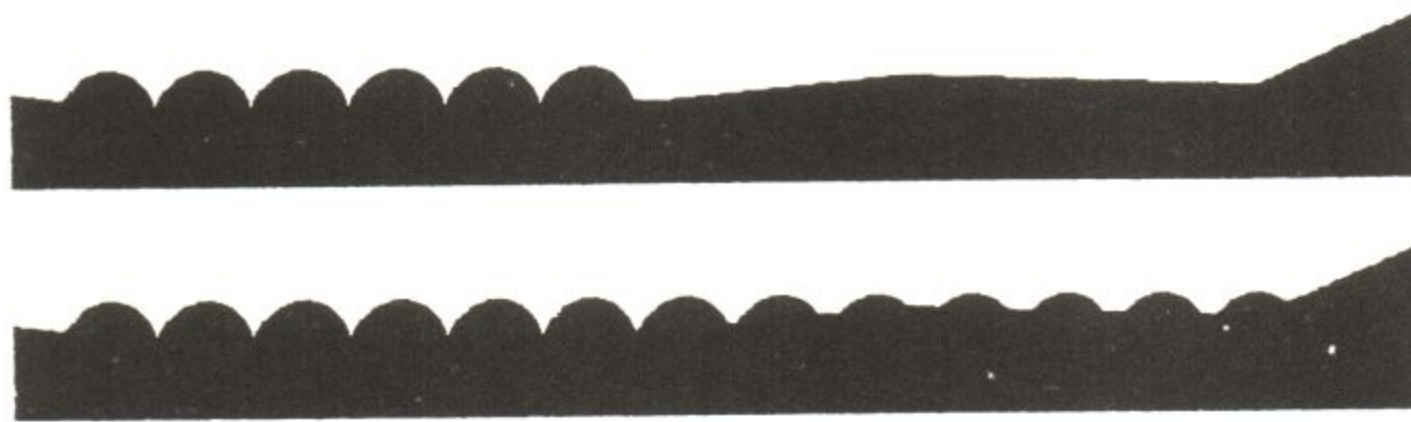
(22)　結果の図を左から右へ動かすプログラムをつくれ.

〔解　答〕

(20) **【プログラム例】**

```
SCREEN 0
CLS 0
DIM a%(500) ―――――――――――――図をとりこむ配列
CIRCLE (50, 200), 20 ――――――┐図の表示
PAINT (50, 200), 7, 7 ―――――┘
```

```
GET (30, 180)-(70, 220), a%———図形のとり込み
CLS 0
LINE (30, 250)-(620, 250) ———バックの表示
LINE -(620, 180)
LINE -(550, 210)
LINE -(400, 205)
LINE -(300, 215)
LINE -(100, 220)
LINE -(30, 210)
LINE -(30, 250)
PAINT (50, 230), 7, 7
FOR i = 50 TO 580 STEP 40 ———円を左から右へ移動
    PUT (i, 200), a%, OR
FOR j = 1 TO 500: NEXT j
NEXT i
END
```

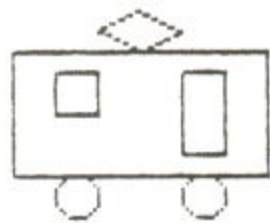
【結　果】

(21)　【プログラム例】

```
SCREEN 0
CLS 0
DIM a%(2700)
CIRCLE (100, 100), 50 ———図形の表示
CIRCLE (115, 80), 10, , , , .5
CIRCLE (85, 80), 10, , , , .5
LINE (103, 65)-(127, 65)
LINE (73, 65)-(97, 65)
LINE (100, 100)-(100, 130)
LINE (80, 140)-(120, 140)
GET (50, 50)-(150, 150), a%———図形のとりこみ
FOR i = 1 TO 3 ———図形の表示
    PUT (150 * i, 200), a%, PSET
NEXT i
END
```

【結　果】

(22)　**【プログラム例】**

```
SCREEN 0
CLS 0
DIM a%(850)
LINE (100, 100)-(160, 130), , B ─┐ 図形の表示
LINE (130, 90)-(140, 95)         │
LINE -(130, 100)                 │
LINE -(120, 95)                  │
LINE -(130, 90)                  │
LINE (110, 105)-(120, 115), , B  │
LINE (140, 105)-(150, 125), , B  │
CIRCLE (115, 135), 5             │
CIRCLE (145, 135), 5 ────────────┘
GET (100, 90)-(160, 140), a% ───── 図形のとり込み
CLS 0
FOR i = 1 TO 50 ─────────────────┐ 図形の表示・消去
    PUT (i * 10, 200), a%, PSET  │
    PUT (i * 10, 200), a%, XOR   │
NEXT i ──────────────────────────┘
END
```

【結　果】

(左から右へ動く)

7章　POINT と DRAW

POINT, DRAW

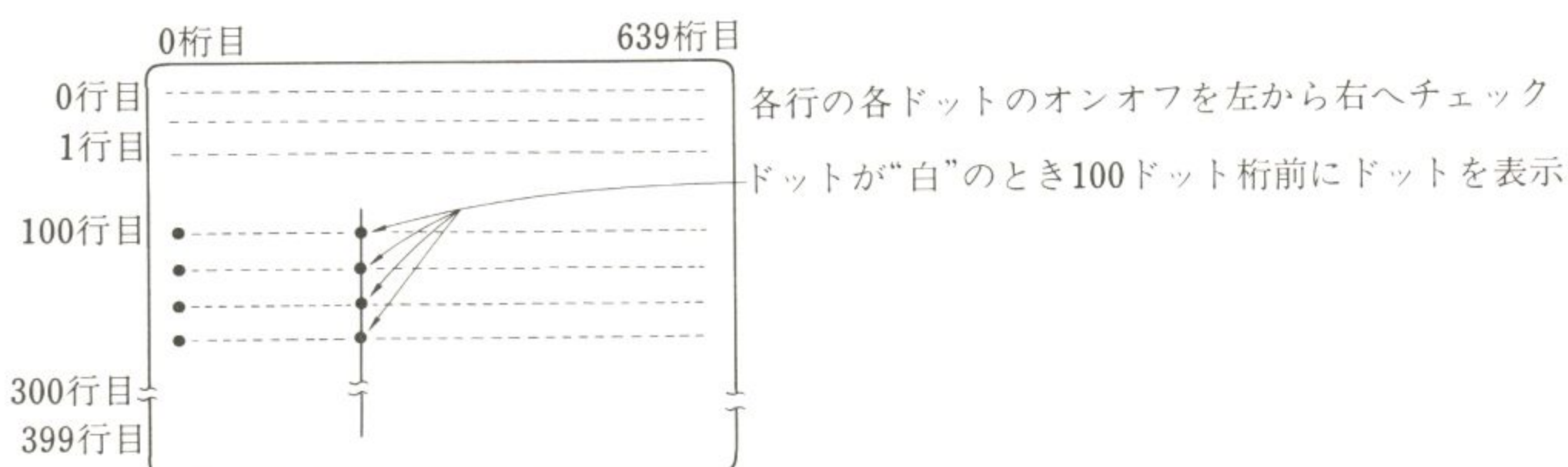
【例題 23】　指定位置の座標値

(100,100)と(100,300)を直線で結び，次に画面を1ドットずつスキャンして，ドットがオンのときそのドット位置の左100ドットの位置のドットをオンにするプログラムをつくれ.

【解　説】　◎　グラフィックカーソルのある位置の座標値

① LINE(100,100)-(100,300)で縦線をかきます.

② 左上(0,0)から(639,0)まで一番上のドット行を左から右へスキャンしていって，ドットがオンのときx方向に-100の位置のドットをオンします.

③ 同様に1行～399行まで同じことをくり返します.

【プログラム例】

```
SCREEN 0
CLS 0
LINE (100, 100)-(100, 300)
FOR i = 0 TO 399
    FOR j = 0 TO 639
        IF POINT(j, i) = 7 THEN PSET (j - 100, i)
    NEXT j
NEXT i
END
```

縦線の表示

0行から399行まで0桁から639桁までドットが白かどうかチェック．白のとき100ドット桁前の1ドットを表示

【結　果】

【例題 24】　DRAW を使った図

DRAW を使って，結果の図をつくれ.

【解　説】　◎　DRAW を使って図をかきます.

① DRAW の後に"アルファベット＋数値"をおいてグラフィックマクロ言語をつくります.

② 機能は次のとおりです．例題を通して順に説明していきます.

　① カーソル移動

　　B，N

　　U，D，L，R，E，F，G，H，M

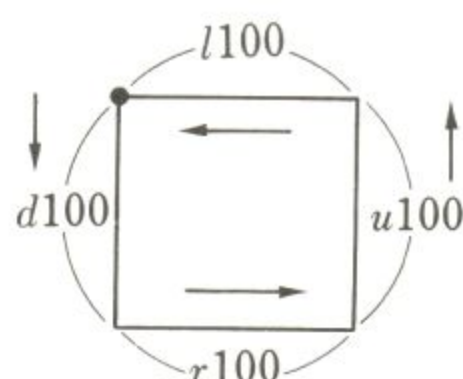
　② 回転，色指定，拡大，縮小

　　A，TA，C，S，P

　③ 部分文字列

③ 3 行目の S2 は拡大率が 2 です．S は拡大縮小を示し，2 は倍率です．2 の部分は 1 から 255 が使えます．実際の値はカーソル移動命令で与えられる数値にこの値を掛けたものです.

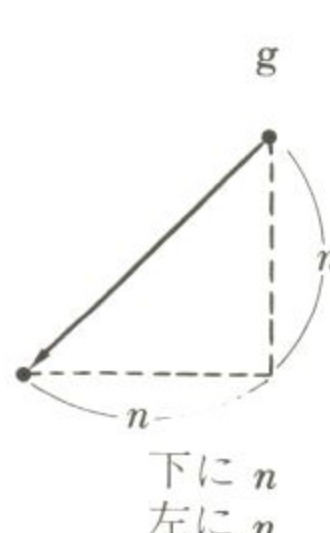
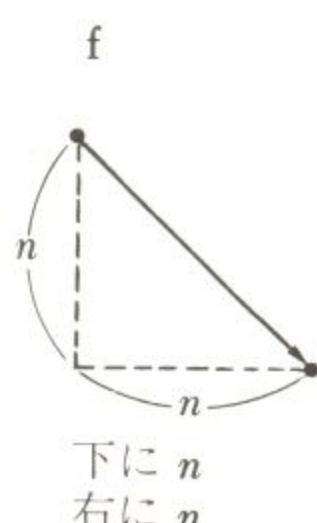
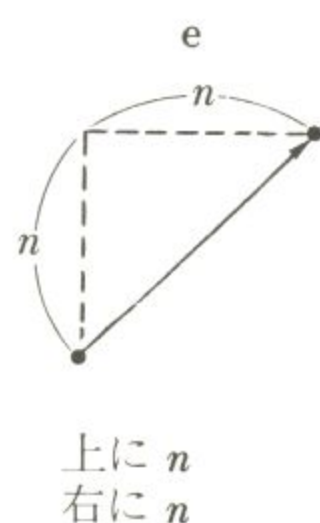
たとえば U50 は上へ 50 ドット移動しますが，S2 が指定してあると上へ 50*2=100 ドット移動します.

④ 4 行目の d100，r100，u100，l100 はおのおの，下，右，上，左へ 100 ドットの移動を示します．ただし，S2 が指定してありますから本例ではその 2 倍の量の移動です.

⑤ 5 行目の bm　100，100 は次の意味です.

B が線をひかずにカーソルのみ移動させます.

Mx，y は座標値(x,y)まで移動します．すなわち，BM100，100 はカーソルのみ(100,100)の位置へ移します.

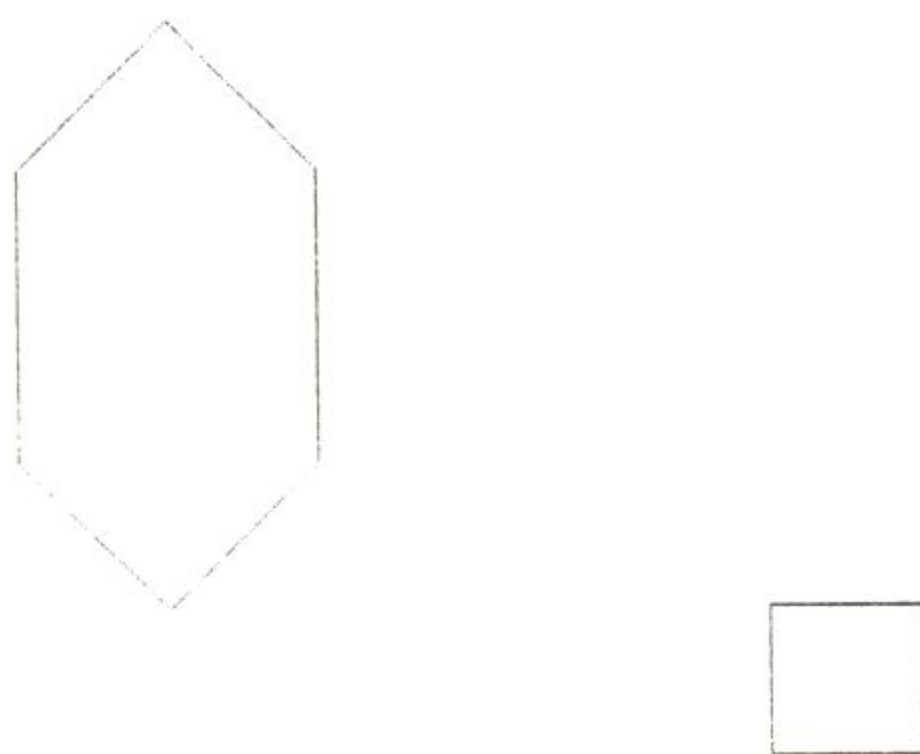
BM+x，+y のように，x，y に符合をつけると現在の位置から相対位置を示します．プラスまたはマイナス x，プラスまたはマイナス y だけ移した位置を示します.

⑥ 7 行目の e30　f30　d60　g30　h30　u60 の e，f，g，h はおのおの次のとおりです.

⑦ 8 行目の nm+100，+0 の n は移動位置まで線をひいた後(0,0)までカーソルをもどします．し

たがって，nm+100，+0 は現在位置から x 方向 100，y 方向 0 の位置までの横線をひいた後，カーソルを(0,0)へもどします．

【プログラム例】

```
SCREEN 0
CLS 0
DRAW "s2"―――――――――――――――2倍の倍率
DRAW "d100 r100 u100 l100"――――四角をかく
DRAW "bm100,100"――――――――――カーソルを移す
DRAW "s5"―――――――――――――――5倍の倍率
DRAW "e30 f30 d60 g30 h30 u60"―――6角形の表示
DRAW "bm0,0 nm+100,+0"――――――(0,0)から横線をひいてカーソルを(0,0)へもどす
END
```

【結　果】

【例題 25】　DRAW を使った図

四角形をかき，次にそれを塗りつぶし，次に四角形を 0 度，90 度，180 度，270 度回転した図をかくプログラムをつくれ．

【解　説】　◎　塗りつぶしと回転

① 3行目4行目で四角形をかきます．

② 5行目で右下へカーソルを移します．

③ 次の C7，P7，7 は次の意味です．

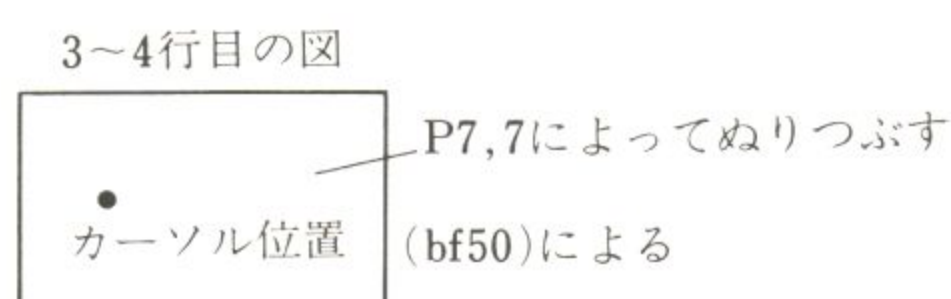
C は色指定です．COLOR，PALETTE および SCREEN 命令と関連します．ここでは 7 によって白を示します．

P は図形の塗りつぶしを意味します．最初の 7 は塗りつぶす色で白，次の 7 は塗りつぶす境界色で白です．すなわち，現在カーソルのある領域で，境界が白線の中を白でぬりつぶします．

3～4行目の図

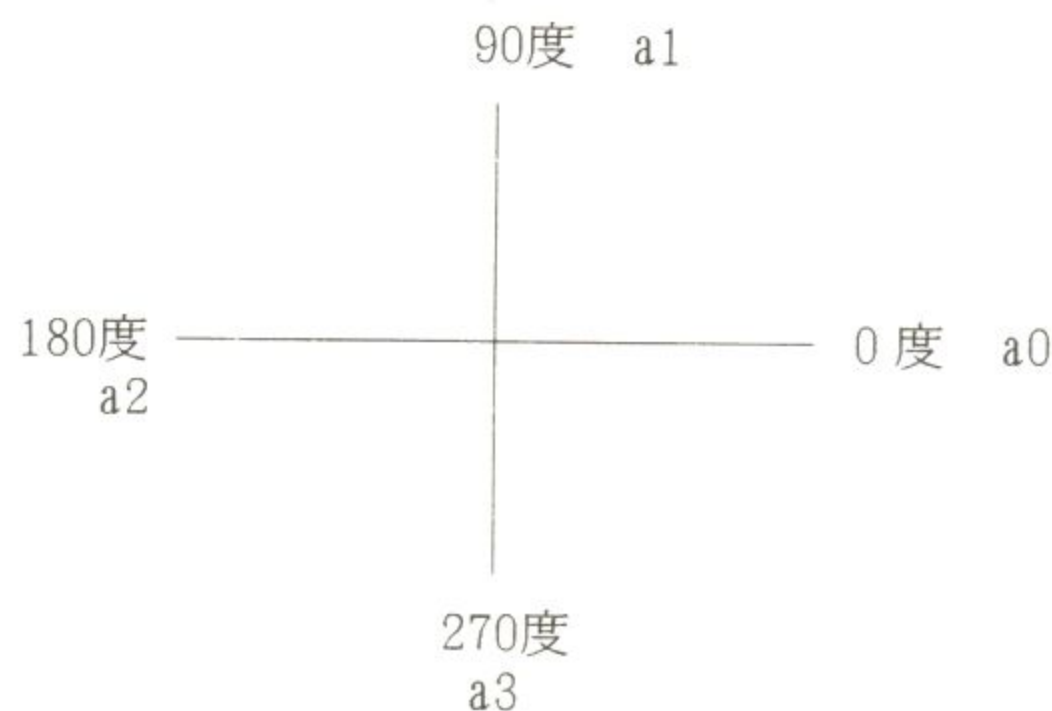
P7,7によってぬりつぶす

カーソル位置

(bf50)による

④ 7行目～10行目の a は回転角を示し数値は角です．角は 0，90，180，270 度でおのおの 0，1，2，3 で表わします．次図のようになります．

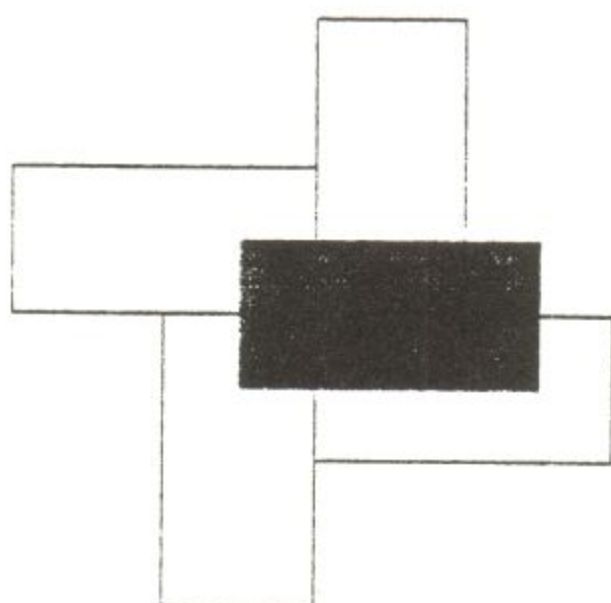
90度　a1

180度　a2　　　0度　a0

270度　a3

⑤　7〜10行目は四角形を回転してかきます.

【プログラム例】

```
SCREEN 0
CLS 0
DRAW "s2"                        ─┐ 四角形をかく
DRAW "d100 r200 u100 l200"       ─┘
DRAW "bf50"                      ── カーソルを四角形の中へ入れる
DRAW "c7 p7,7"                   ── 塗りつぶし
DRAW "a0 d100 r200 u100 l200"    ─┐ 回転した四角形を4つかく
DRAW "a1 d100 r200 u100 l200"     │
DRAW "a2 d100 r200 u100 l200"     │
DRAW "a3 d100 r200 u100 l200"    ─┘
END
```

【結　果】

【例題 26】　DRAW を使った図

四角形をかき次に60度反時計方向と180度時計方向へ回転した図でかくプログラムをつくれ.

【解　説】　◎　任意の角の回転をします.

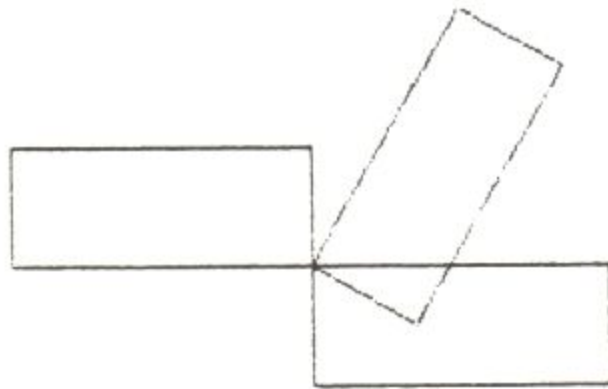
①　3〜4行目で四角形をかきます.

②　5行目と6行目の ta は回転を示します. 数値は-360から360までの整数で, 正のとき反時計まわり, 負のとき時計まわりとなります.

　したがって5行目の ta60 は反時計まわりに60度, 6行目の ta-180 は時計まわりに180度の回転を示します.

【プログラム例】

```
SCREEN 0
CLS 0
DRAW "s2"────────────────────┐── 四角形の表示
DRAW "d100 r200 u100 l200"───┘
DRAW "ta60 d100 r200 u100 l200"──── 60 度反時計方向へ回転した図
DRAW "ta-180 d100 r200 u100 l200"─── 180 度時計方向へ回転した図
END
```

【結　果】

【例題 27】　**DRAW を使った図**

DRAW を使って，四角形をかき，次に長さが 2 倍，3 倍の図をかくプログラムをつくれ．

【解　説】　◎　拡大図をつくります．

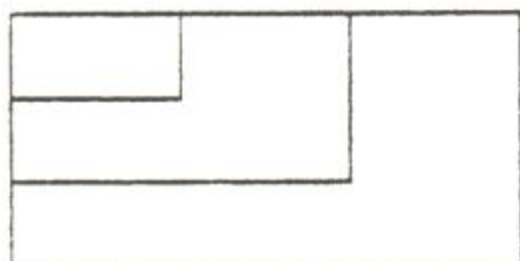
① 3 行目で縦 100，横 200 の四角形をつくります．

② 4 行目で S2 の指定により，縦，横とも長さが 2 倍の図をつくります．

③ 5 行目で S3 の指定により，縦，横とも長さが 3 倍の図をつくります．

【プログラム例】

```
SCREEN 0
CLS 0
DRAW "s1 d100 r200 u100 l200"──── 四角形をかく
DRAW "s2 d100 r200 u100 l200"──── 2 倍の四角形をかく
DRAW "s3 d100 r200 u100 l200"──── 3 倍の四角形をかく
END
```

【結　果】

【例題 28】　**DRAW を使った図**

DRAW を使って【結　果】の図をかけ．

【解　説】　◎　部分文字列コマンドを使います．

① 3 行目と 6 行目の組合わせは次のようになります．

3 行目　a1$=　"S2"　　　　　　　　　→DRAW　"S2"

6行目　DRAW　"x"+VARPTR$(a1$)

② 同様に5行目と7行目の組合わせは次のとおりになります.

5行目　a3$="d100r100u100l100"　　→DRAW"d100　r100　u100　l100"

7行目　DRAW"x"+VARPTR$(a3$)

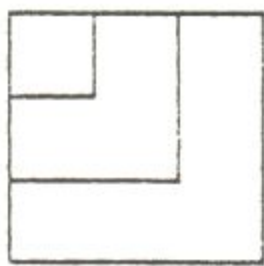
③ 10行目と11行目の組合わせは次のようになります.

10行目　Z=1　　→DRAW　"S1"

11行目　DRAW　"S="+VARPTR$(Z)

【プログラム例】

```
SCREEN 0
CLS 0
a1$ = "s2"                       ─┐変数に文字列を代入
a2$ = "s3"                        │
a3$ = "d100 r100 u100 l100"      ─┘
DRAW "x" + VARPTR$(a1$)          ─┐DRAW文
DRAW "x" + VARPTR$(a3$)           │
DRAW "x" + VARPTR$(a2$)           │
DRAW "x" + VARPTR$(a3$)          ─┘
z = 1                            ── 変数に数値を代入
DRAW "s=" + VARPTR$(z)           ─┐DRAW文
DRAW "x" + VARPTR$(a3$)          ─┘
END
```

【結　果】

〔演習問題〕

(23)　画面左上に"日"をかき，画面のドットをチェックして，表示のとき100ドット桁プラス，200ドット桁プラスの位置のドットを表示することで"日"を右に2つコピーするプログラムをつくれ.

(24)　"S2"を指定して縦100，横300の長方形をかき，次にa0，a1，a2，a3を指定して回転した図をつくるプログラムをつくれ.

(25)　結果の図をかくプログラムをつくれ.

(26)　結果の図をかくプログラムをつくれ.

(27)　結果の図をかくプログラムをつくれ.

〔解　答〕

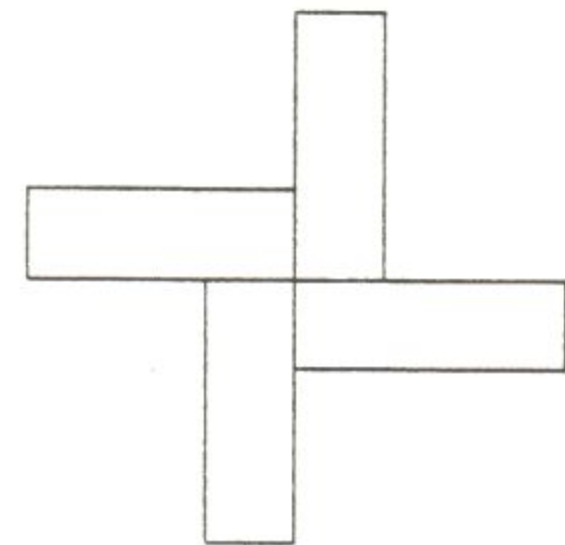
(23)　**【プログラム例】**

```
SCREEN 0
CLS 0
LINE (10, 10)-(50, 50), , B ──────┐─"日"の表示
LINE (10, 30)-(50, 30) ───────────┘
FOR i = 0 TO 60 ──────────────────┐─各ドットをチェックして
   FOR j = 0 TO 60                │ 表示してあるとき
      IF POINT(i, j) = 7 THEN     │ 100ドット桁プラス, 200ドット桁
         PSET (i + 100, j)        │ プラスの位置のドットを表示
         PSET (i + 200, j)        │
      END IF                      │
   NEXT j                         │
NEXT i ───────────────────────────┘
END
```

【結　果】

(24)　**【プログラム例】**

```
SCREEN 0
CLS 0
DRAW "s2"──────────────────────────┐─長方形の表示
DRAW "d100 r300 u100 l300"─────────┘
DRAW "a0 d100 r300 u100 l300"──────┐─回転した図の表示
DRAW "a1 d100 r300 u100 l300"      │
DRAW "a2 d100 r300 u100 l300"      │
DRAW "a3 d100 r300 u100 l300"──────┘
END
```

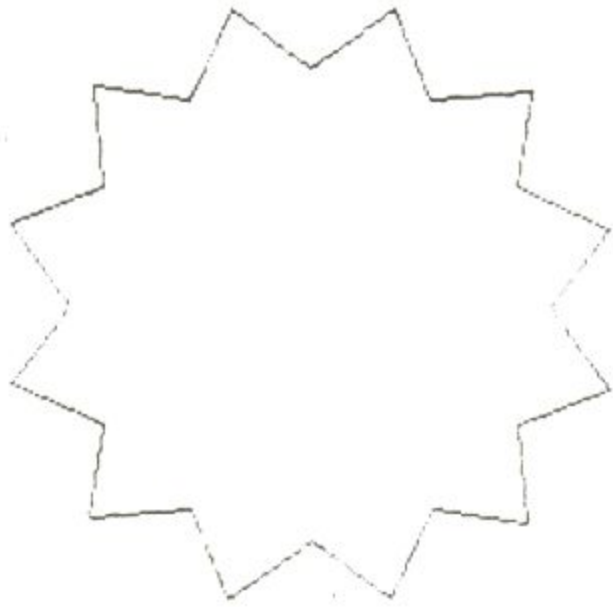
【結　果】

(25)　**【プログラム例】**

```
SCREEN 0
CLS 0
DRAW "bm300,100"─────────────カーソルを(300,100)
DRAW "s2"                     へ移動
DRAW "f100 r100 g100 d100"───┐─図の表示
DRAW "l100 h100 u100 e100"───┘
END
```

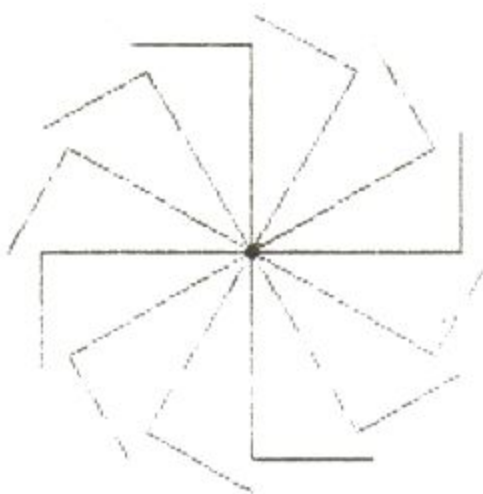
【結　果】

(26)　【プログラム例】

```
SCREEN 0
CLS 0
DRAW "s1"
DRAW "bm300,300"──────────カーソルを(300,300)へ移動
FOR i = 0 TO 360 STEP 30 ──┬─30度ずつ回転した図を表示
    DRAW "ta=" + VARPTR$(i)  │
    DRAW "r100 u100"         │
NEXT i ─────────────────────┘
END
```

【結　果】

(27)　【プログラム例】

```
SCREEN 0
CLS 0
DRAW "s1"
DRAW "bm300,200"──────────カーソルを(300,200)へ移動
FOR i = 0 TO 360 STEP 30 ──┬─30度ずつ回転した図を表示
    DRAW "ta=" + VARPTR$(i)  │
    DRAW "r200 u100"         │
    DRAW "bm300,200"         │
NEXT i ─────────────────────┘
END
```

【結　果】

IV……応用編

1章 N_{88}BASIC（86）との互換

【応用問題1】 表 示

N_{88}BASIC(86)でつくった次のプログラムをQuick BASICで実行せよ.

```
10 PRINT "ｸｲｯｸﾍﾞｰｼｯｸ"
20 END
```

【解 説】 ◎ N_{88}BASIC(86)のプログラムをQuick BASICで実行します.

① N_{88}BASIC(86)を使って，問題のプログラムをつくります．これをアスキーファイルとして格納します.

save "PRAC-1.BAS", a ↵ によってアスキーセーブします.

② Quick BASICを走らせます．DIRによってPRAC-1.BASがあることを確認します.

```
RRR-82   BAS   RRR-85   BAS   HAIRETSU BAS   HAIRETSU IN    RRR-86   BAS
PAI      IN    RRR-87   BAS   RRR-88   BAS   KAMOKU   IN    RRR-89   BAS
RRR-90   BAS   RRR-95   BAS   RRR-96   BAS   RRR-97   BAS   RRR-98   BAS
RENSHU1  BAS   PRAC-1   BAS
     142 個のファイルがあります.
  651264 バイトが使用可能です.
```

③ ファイルメニューを選択し，O，PRAC-1. ↵ によってファイルPRAC-1.BASを読み込みます.

```
                          PRAC-1.BAS
10 PRINT "ｸｲｯｸﾍﾞｰｼｯｸ"
20 END
```

④ このプログラムはこのまま実行できます．f･5 によってプログラムを実行します.

```
ｸｲｯｸﾍﾞｰｼｯｸ
```

【プログラム例】

```
10 PRINT "ｸｲｯｸﾍﾞｰｼｯｸ"
20 END
```

【応用問題 2】 桁・行の指示

次のプログラムを N_{88}BASIC(86)でつくり，これを Quick BASIC で実行せよ．

```
10 CLS
20 FOR I = 1 TO 10
30      LOCATE I, 10: PRINT "A"
40 NEXT I
50 END
```

【解 説】 ◎ N_{88}BASIC(86)でつくったプログラムを Quick BASIC で一部変更して実行します．

① N_{88}BASIC(86)を使って問題のプログラムをつくります．

② これをアスキーセーブします．

save "PRAC-2.BAS", a です．

③ Quick BASIC を走らせファイルメニューから O，PRAC-2 ↵ によりこのプログラムを読み出します．

```
10 CLS
20 FOR I = 1 TO 10
30      LOCATE I, 10: PRINT "A"
40 NEXT I
50 END
```

④ f･5 で実行すると次のように縦に"A"を表示します．

```
A
A
A
A
A
A
A
A
A
A
```

⑤ 元のプログラムでは横に"A"を表示するものですから，縦横が逆です．これは LOCATE x, y が N_{88}BASIC(86)では x が桁，y が行を示すのに対し，Quick BASIC では x が行，y が桁を示すからです．したがって，桁と行を逆に修正します．

【プログラム例】

```
10 CLS
20 FOR I = 1 TO 10
30      LOCATE 10, I: PRINT "A"          LOCATE i,10 を LOCATE 10,i に変更
40 NEXT I
50 END
```

【結 果】

```
AAAAAAAAAA
```

【応用問題3】　グラフィック

次のプログラムをN_{88}BASIC(86)でつくり，次にこれをQuick BASICで実行せよ．

```
10 SCREEN 2, 0, 0, 1
20 CLS 3
30 LINE (200, 50)-(400, 250), , B
40 LINE (200, 50)-(400, 250)
50 LINE (200, 250)-(400, 50)
60 LINE (100, 250)-(500, 350), , BF
70 END
```

【解　説】　◎　N_{88}BASIC(86)でつくったグラフィックスをQuick BASICで実行します．

① 問題のプログラムをN_{88}BASIC(86)でつくります．

② これをアスキーセーブします．

save　"PRAC-3.BAS"，　a です．

③ Quick BASICでこれを読み出します．ファイルメニューから O ，PRAC-3.BAS ↵ です．

④ これをそのまま f･5 とするとエラーとなります．

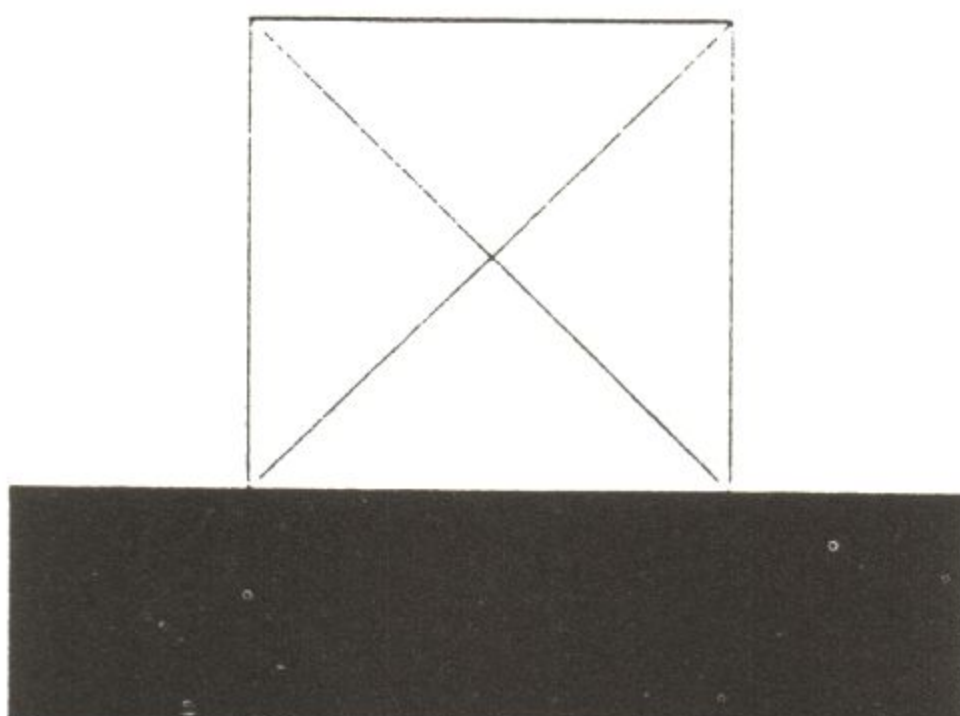
SCREENは，N_{88}BASIC(86)とQuick BASICでは，使い方が違います． Quick BASICのSCREEN文の2番目は8086/v30，80286をCPUとするマシンでは指定できません．

⑤ 行番号10と20をつぎのように修正します．

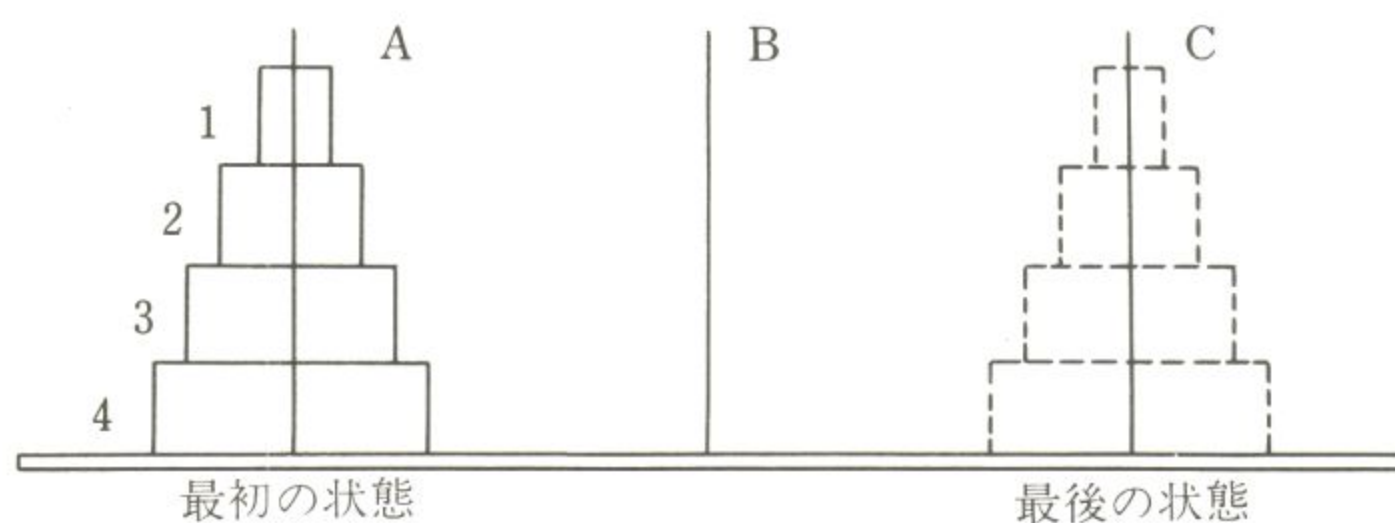
PRAC-4.BAS

```
10 SCREEN 0
20 CLS
30 LINE (200, 50)-(400, 250), , B
40 LINE (200, 50)-(400, 250)
50 LINE (200, 250)-(400, 50)
60 LINE (100, 250)-(500, 350), , BF
70 END
```

⑥ f･5 で実行します．

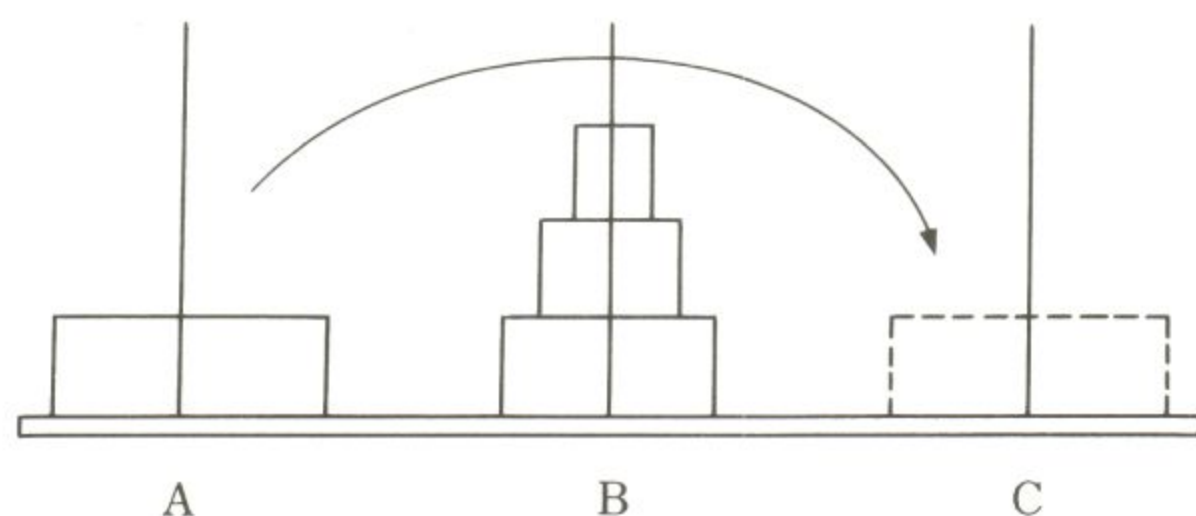
【結　果】

2章 再帰プログラム

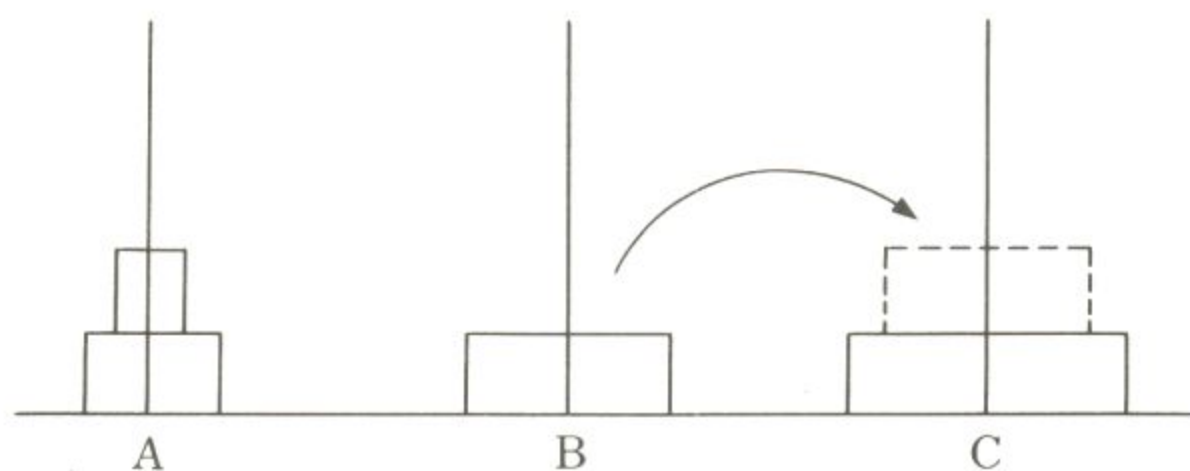
【応用問題4】 ハノイの塔

再帰プログラムを利用してハノイの塔を解くプログラムをつくれ．ここでの例は次のような問題とします．

1～4の円板（円筒），A～Cの柱があります．Aの柱に大きい円板を下にして1～4の円板をさしこみます．これを1回に1つの円板を動かし，小さい円板の上に大きい円板が乗らないことを条件にしてCの柱に大きい順に並べなおしてください．

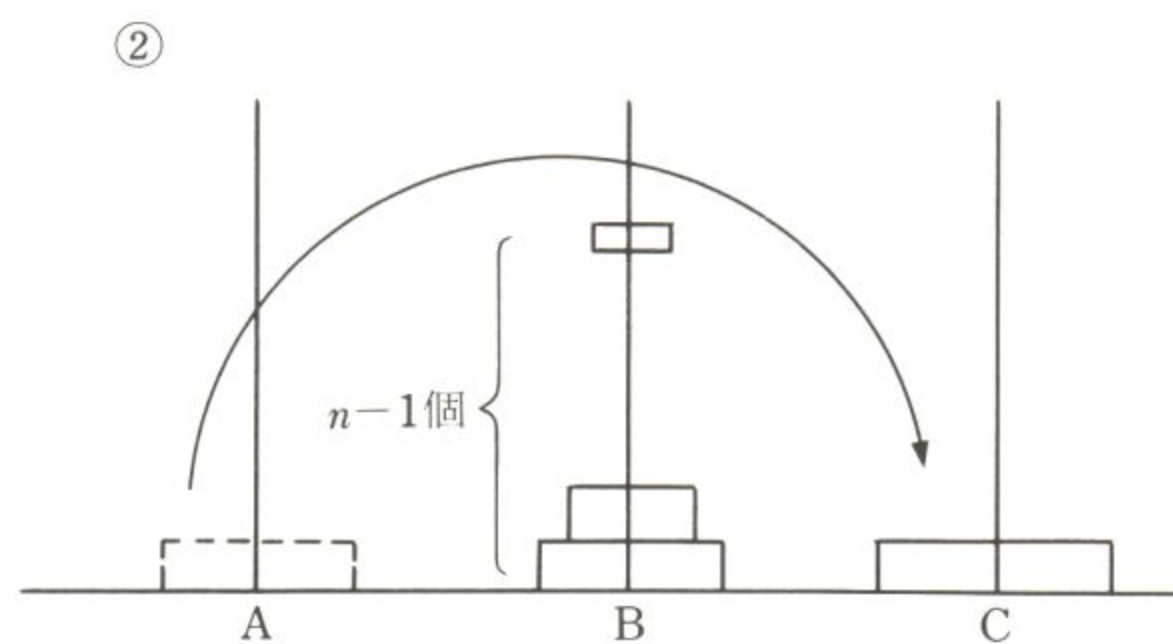
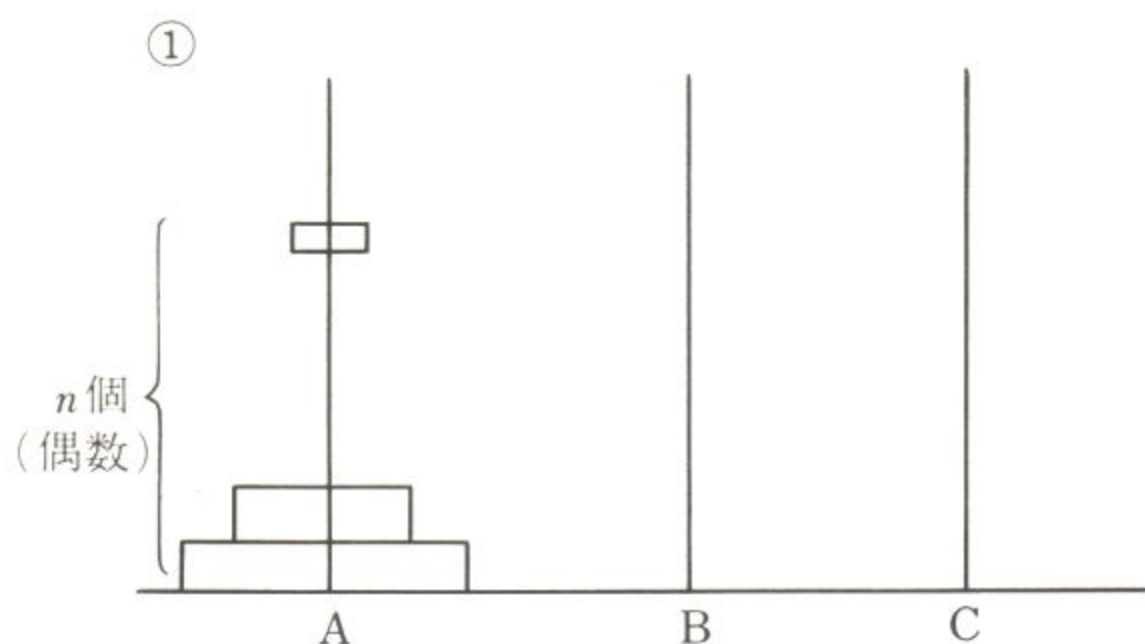
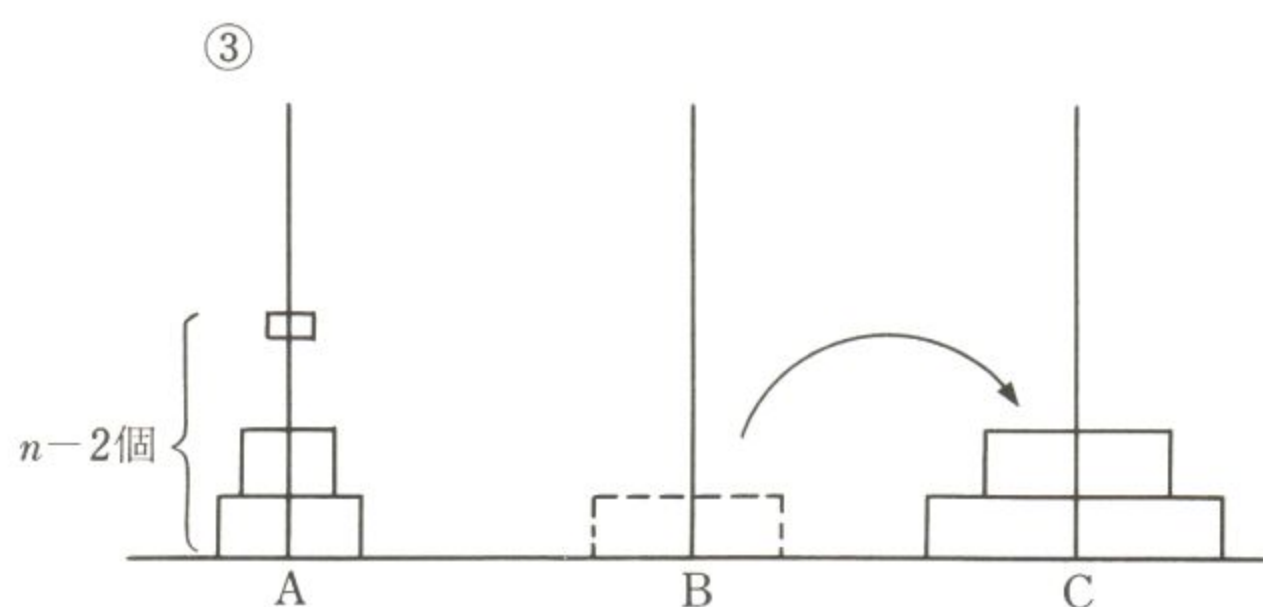
【解 説】 ◎ ハノイの塔の問題を再帰プログラムで解きます．

① ハノイの塔の問題は複雑そうですが，再帰という考えを入れると簡単に解けます．この考え方をまず説明します．説明をみる前に自分でも実際にやってみてください．

② 一番大きい円板がAの柱からCの柱へ動くときの状態を考えてください．次のようになるはずです．

"Bに残りが大きい順に並んでいる"ことがポイントです．

③ 次に残りの円板のうち最大のものが，Cの円板の上へ乗る直前の状態は次のとおりです．

Aに残りが大きい順に並んでいることがポイントです．

④ すでに気づいたとおり，n個の中の最大のものを動かすときは動かす先はnより大きいものが順

に並んでいるか何もない状態であり，他の柱はn-1個の円板が大きい順に並んでいます．ということはn個のものを移すことはn-1個のものを移すことと同じことであり，n-1個のものを移すことはn-2個のものを移すのと同じことです．

n個のものを移すことは最大の円板を移した後はn-1個のものをとなりの柱から移すのと同じである．このくり返しとなります．

⑤　再帰的に考えると次のようになります．

```
SUB hanoi(x%,a$,b$,c$)            …x%を4，a$を"A"，b$を"C"，c$を"B"としてhanoiを呼び出します．
   1F x%>0 THEN
      hanoi x%-1,a$,c$,b$         ……n-1個の円板をaからbへ
      PRINT x%;"ノエンバンオ";a$"カラ";b$;"へ"     ……最大のものをaからcへ
      hanoi x%-1,c$,b$,a$
   END IF
END SUB
```

【プログラム例】

```
DECLARE SUB hanoi (x%, a$, b$, c$)
hanoi 4, "A", "C", "B"
END

SUB hanoi (x%, a$, b$, c$)
    IF x% > 0 THEN
        hanoi x% - 1, a$, c$, b$
        PRINT x%; " ﾉ ｴﾝﾊﾞﾝ ｵ "; a$; " ｶﾗ "; b$; " ﾍ"
        hanoi x% - 1, c$, b$, a$
    END IF
END SUB
```

【結　果】

```
1  ノ エンバ゛ン オ A カラ B ヘ
2  ノ エンバ゛ン オ A カラ C ヘ
1  ノ エンバ゛ン オ B カラ C ヘ
3  ノ エンバ゛ン オ A カラ B ヘ
1  ノ エンバ゛ン オ C カラ A ヘ
2  ノ エンバ゛ン オ C カラ B ヘ
1  ノ エンバ゛ン オ A カラ B ヘ
4  ノ エンバ゛ン オ A カラ C ヘ
1  ノ エンバ゛ン オ B カラ C ヘ
2  ノ エンバ゛ン オ B カラ A ヘ
1  ノ エンバ゛ン オ C カラ A ヘ
3  ノ エンバ゛ン オ B カラ C ヘ
1  ノ エンバ゛ン オ A カラ B ヘ
2  ノ エンバ゛ン オ A カラ C ヘ
1  ノ エンバ゛ン オ B カラ C ヘ
```

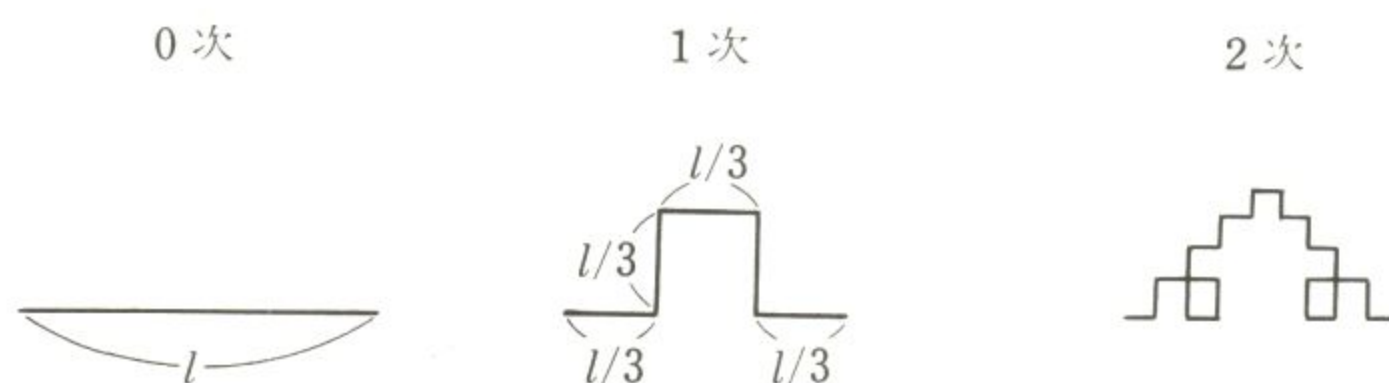
【応用問題5】　ステッチ

再帰を使って，結果のステッチ模様をかくプログラムをつくれ.

【解　説】

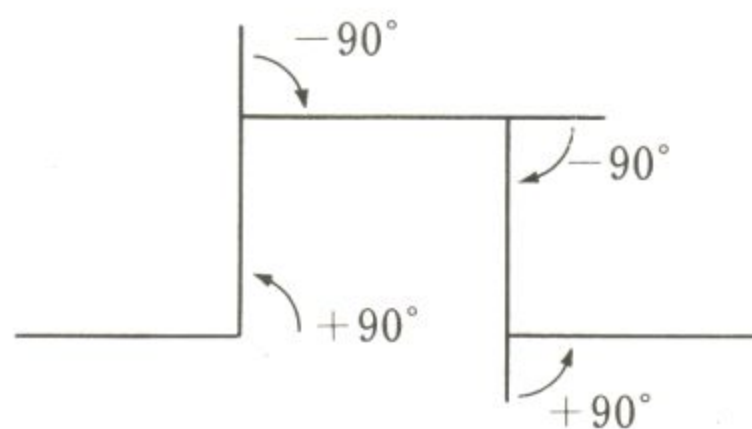
① クロスステッチはコッホ曲線を使ってかきます．コッホ曲線は次のようにつくります.

0次　1次　2次

$l/3$　$l/3$　l　$l/3$　$l/3$

0次は直線長さ1
1次は左から1/3のところで1/3の正方形（ただし，下辺はなし）をつくる
2次は1次の各直線部で1/3のところすなわち1/9のところで正方形をつくる

② 1次のコッホ曲線は0次のコッホ曲線を5本かいたものです．2次のコッホ曲線は1次のコッホ曲線を5本かいたものです．一般化するとn次のコッホ曲線はn-1次のコッホ曲線を5本かけばよいことになります.

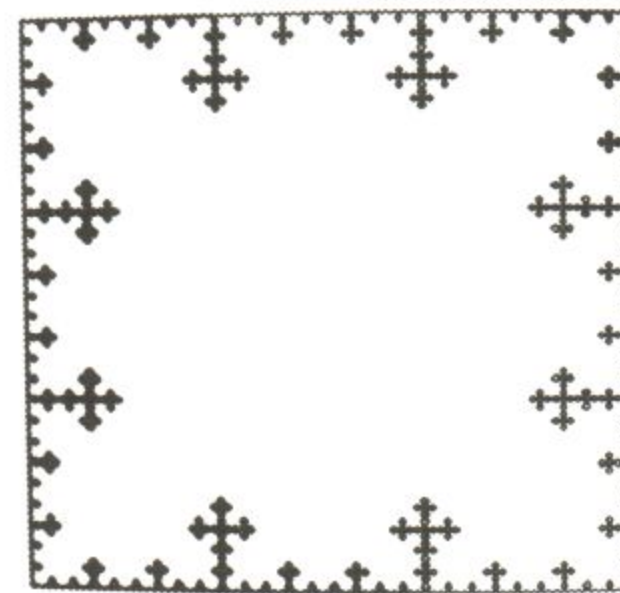
このとき正方形の方向は次のように+90，-90，-90，+90度です.

−90°　−90°　+90°　+90°

したがって，プログラム例のサブルーチンstechがコッホ曲線をかく再帰呼び出しを行うルーチンです.

【プログラム例】

```
DECLARE SUB stech (n!, l)
DECLARE SUB move (l)
DECLARE SUB kaku (deg!)
COMMON SHARED xx, yy, a
```

```
SCREEN 0
CLS 0
xx = 300
yy = 350
a = 45
n = 4
l = 3
FOR i = 1 TO 4
    stech n, l
    kaku 90
NEXT i
END

SUB kaku (deg)
    a = a + deg
    a = a MOD 360
END SUB

SUB move (l)
    x = l * COS(3.14159 / 180 * a)
    y = l * SIN(3.14159 / 180 * a)
    LINE (xx, yy)-(xx + x, yy - y)
    xx = xx + x
    yy = yy - y
END SUB

SUB stech (n, l)
    IF n = 0 THEN
       move l
    ELSE
       stech n - 1, l
       kaku -90
       stech n - 1, l
       kaku 90
       stech n - 1, l
       kaku 90
       stech n - 1, l
       kaku -90
       stech n - 1, l
    END IF
END SUB
```

【結　果】

3章　円弧を使ったグラフ

【応用問題6】　円帯グラフ

下の表のデータを円帯グラフとするプログラムをつくれ.

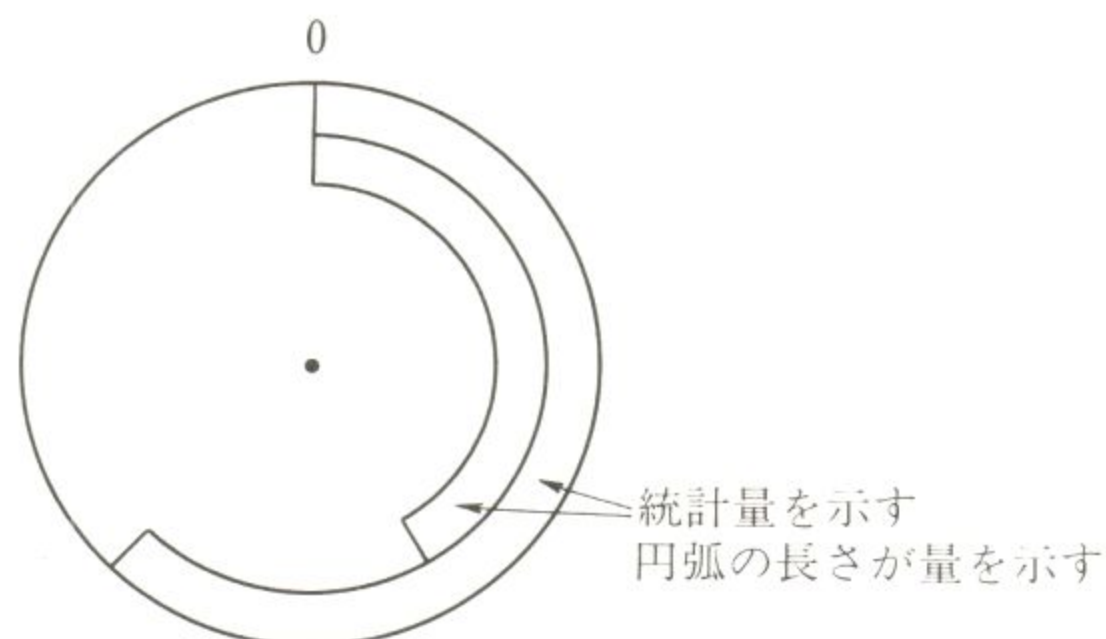
1987年商品別売上高

品　　名	売上高（億円）
家　電	250
モーター	220
コントローラー	150
エレクトロニクス	360

【解　説】　◎　円帯グラフをつくります.

①　下図のように円周上の円弧の長さで統計量を表わすグラフを円帯グラフと名づけます.

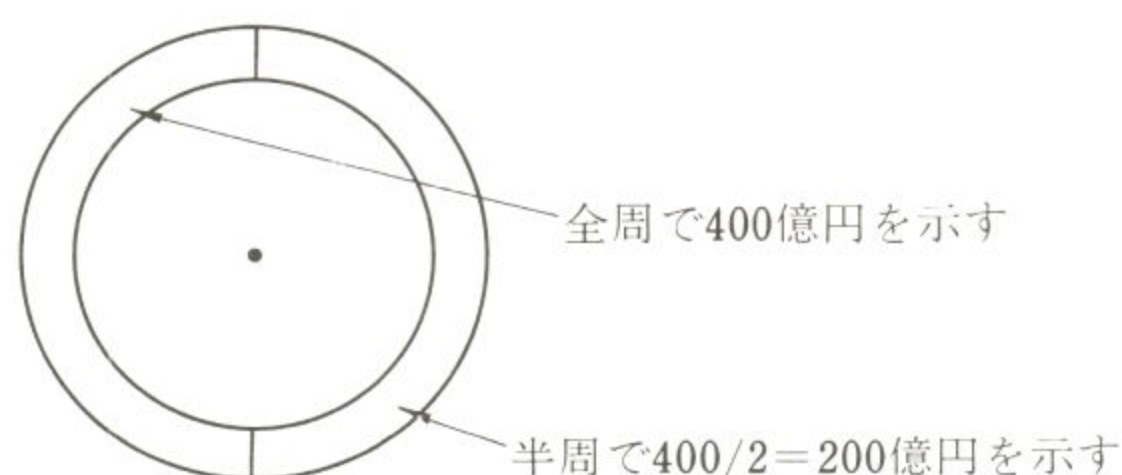
0

統計量を示す
円弧の長さが量を示す

円弧の長さは360度で比率なら100%，量ならグラフで表示できる最大量とします.

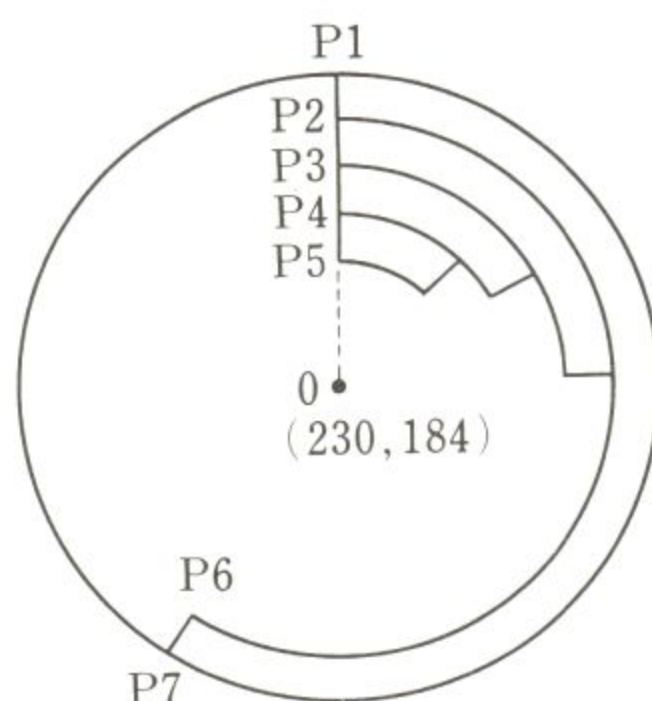
②　4つの統計量を大きさの順に並べなおし，量の多いものを外周側へおきます.

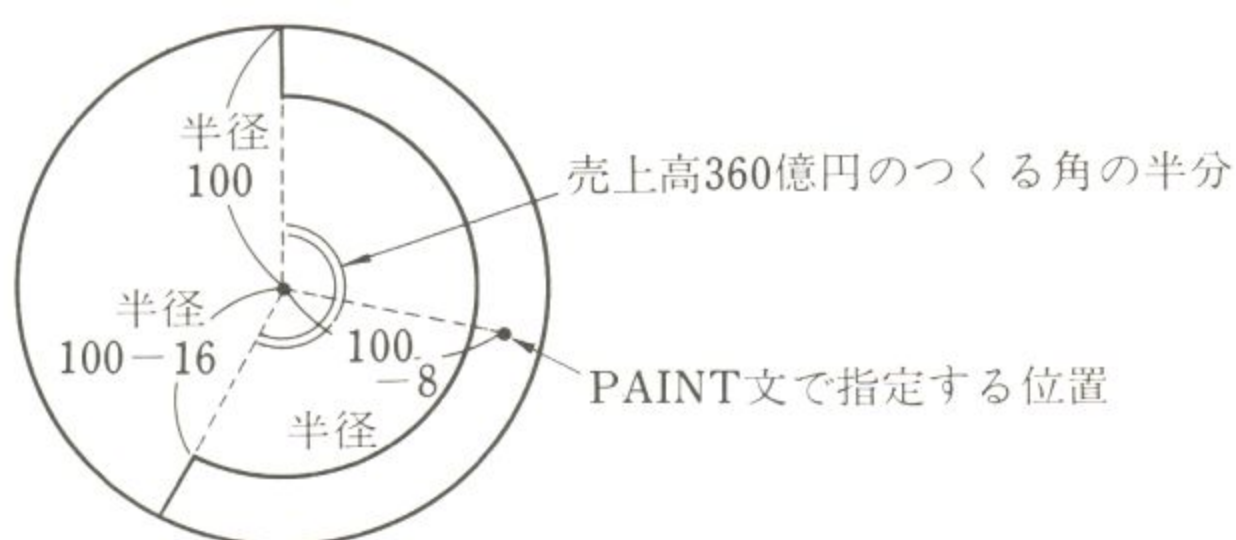
③　売上高は360億円が最大ですから，ここでは全周すなわち360度の場合400億円を示すものとします.

全周で400億円を示す

半周で400/2＝200億円を示す

④　したがって，X億円の売上はX/400*2*3.14159ラジアンで表示できます．400億円の場合2*3.14159ラジアンすなわち360度です.

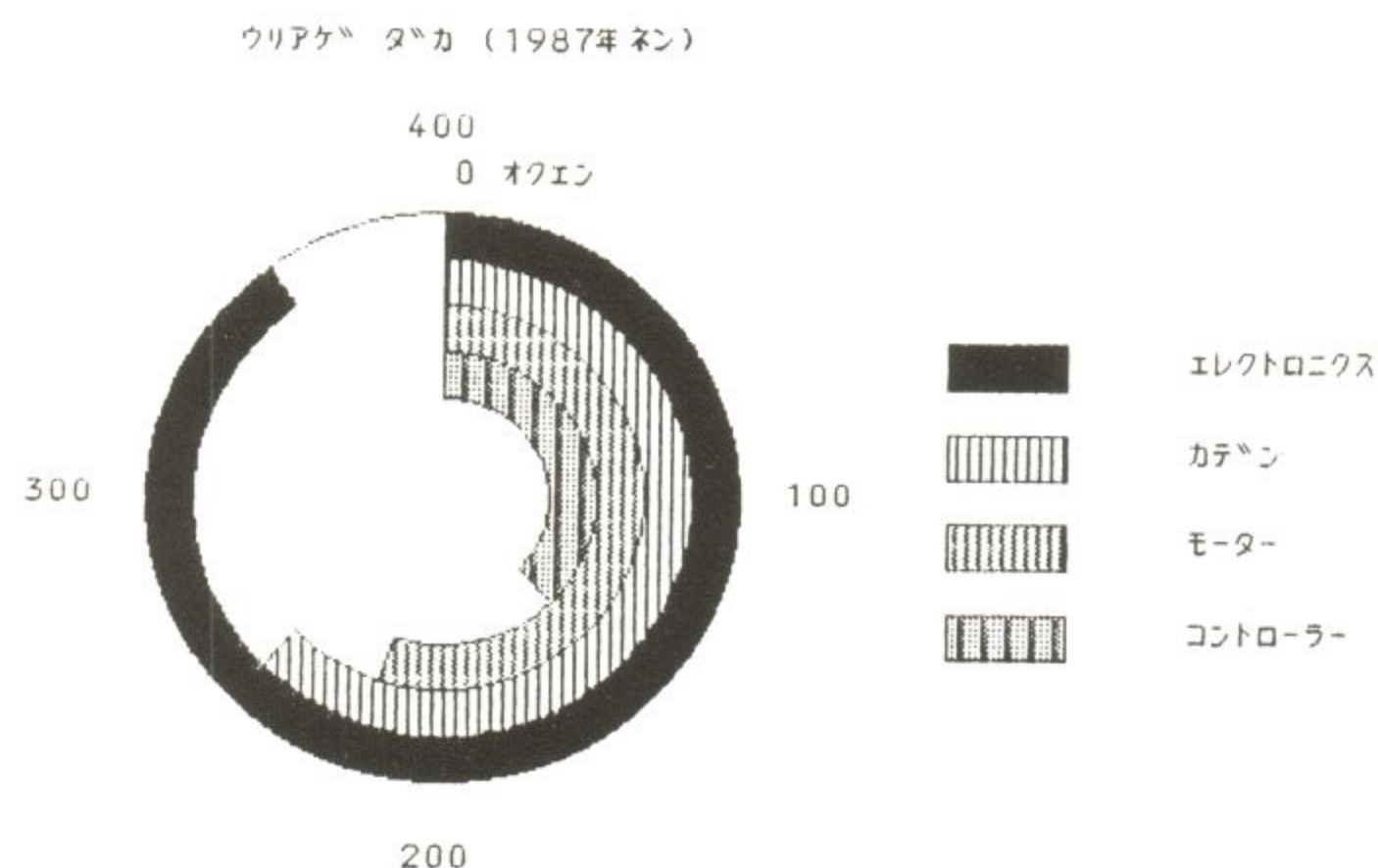
⑤　図の表示は次のようになります.

P1
P2
P3
P4
P5
0
(230,184)
P6
P7

① 中心を(230,184)，半径を 100 とする円をかく
② 半径上の P1 と P5 を結ぶ．P1 は半径 100 の円周上の点で時計の 12 時の位置です．P5 は P1 と中心を結ぶ半径上の点で P1 と P5 の長さは 4 本の帯の幅です．今回は 1 本の帯の幅は 16 ドットとしています．したがって，P1-P5 は Y 方向 16*4=64 ドットです．
③ P2-P6 の円弧をかきます．中心を (230,184) とし，半径は (100-16) ドットで，P2-0-P6 のつくる角度はエレクトロニクスの売上高 360/400*2*3.14159 ラジアンです．
④ P6 と P7 を結びます．P7 は中心を (230,184) とし，半径を 100 とする円周上で，P1-0-P7 のつくる角がエレクトロニクスの売上高 360/400*2*3.14159 ラジアンです．
⑤ 帯の中の 1 点を指定して PAINT 文でハッチングします．帯の中の 1 点は中心を (230,184) とし，半径を (100-8) ドット，角度をエレクトロニクスの売上高 360/400/2*2*3.14159 ラジアンの位置です．

半径
100
売上高360億円のつくる角の半分
半径
100－16
100
－8
PAINT文で指定する位置
半径

【プログラム例】

```
DECLARE SUB aa ()
DECLARE SUB bb ()
COMMON SHARED a$(), a(), b$()
DIM a$(1 TO 4), a(1 TO 4), b$(1 TO 4)
SCREEN 0
CLS 0
DEF fnx (c, r, x) = c + r * SIN(x)―――― 円周上の点の X 座標値の定義
DEF fny (c, r, y) = c - r * COS(y)―――― 円周上の点の Y 座標値の定義
DEF fnd (d) = d / 400 * 2 * 3.14159 ―――― 売上高に相当する中心角を求める式の定義
b$(1) = CHR$(&HFF) ―――――――――――――――― ハッチングパターンの登録
b$(2) = CHR$(&H11)
b$(3) = CHR$(&H22) + CHR$(&H11)
b$(4) = CHR$(&HC) + CHR$(&HAA)
FOR i = 1 TO 4 ――――――――――――――――――― 商品名と売上の読み込み
    READ a$(i), a(i)
NEXT i
FOR i = 1 TO 3 ――――――――――――――――――― 売上高の大きい順の並べかえ
    FOR j = i + 1 TO 4
        IF a(i) > a(j) THEN GOTO a:
        SWAP a$(i), a$(j)
        SWAP a(i), a(j)
a:  NEXT j
NEXT i
```

```
CALL aa ──────────── グラフ準備ルーチンへ分岐
CALL bb ──────────── グラフ作成ルーチンへ分岐
END
DATA ｶﾃﾞﾝ,250,ﾓｰﾀ,220,ｺﾝﾄﾛｰﾗ,150
DATA ｴﾚｸﾄﾛﾆｸｽ,360

SUB aa ──────────── グラフ準備ルーチン
    LOCATE 1, 20: PRINT "ｳﾘｱｹﾞﾀﾞｶ(1987ﾈﾝ)"──── 標題
    FOR i = 128 TO 224 STEP 32 ──────── 凡例のパターン
        LINE (400, i)-(440, i + 16), , B
        PAINT (420, i + 8), b$((i - 96) / 32)
    NEXT i
    FOR i = 8 TO 14 STEP 2 ──────── 凡例の商品名
        LOCATE i, 60: PRINT a$(i / 2 - 3)
    NEXT i
    LOCATE 4, 29: PRINT "0 ｵｸｴﾝ"──────── 単位，売上高の値
    LOCATE 3, 26: PRINT 400
    LOCATE 11, 42: PRINT 100
    LOCATE 19, 26: PRINT 200
    LOCATE 11, 10: PRINT 300
END SUB

SUB bb ──────────── グラフの作成ルーチン
    CIRCLE (230, 184), 100 ──────── 円
    LINE (230, 84)-(230, fny(184, 100 - 4 * 16, fnd(0))) ──── 基準線
    FOR i = 1 TO 4 ──────── 4データの帯
        PSET (fnx(230, 100 - i * 16, fnd(0)), fny(184, 100 - i * 16, fnd(0)))
            FOR q = 0 TO fnd(a(i)) + .03 STEP .05 ──── 売上高に相当する円弧
                x1 = fnx(230, 100 - i * 16, q)
                y1 = fny(184, 100 - i * 16, q)
                LINE -(x1, y1)
            NEXT q
        x2 = fnx(230, 100 - (i - 1) * 16, fnd(a(i))) ──── 帯の最終端の直線
        y2 = fny(184, 100 - (i - 1) * 16, fnd(a(i)))
        LINE -(x2, y2)
        x3 = fnx(230, 100 - (i - 1) * 16 - 8, fnd(a(i) / 2)) ──── ハッチング
        y3 = fny(184, 100 - (i - 1) * 16 - 8, fnd(a(i) / 2))
        PAINT (x3, y3), b$(i)
    NEXT i
END SUB
```

【結 果】

【応用問題 7】　両側円帯グラフ

次表を両側円帯グラフとするプログラムをつくれ.

男女別成績

課目	男	女
英語	85	70
国語	75	85
社会	80	70
数学	88	75

【解　説】　◎　両側円帯グラフをつくります.

①　【応用問題 6】でつくった帯グラフを応用し，基準線から両側に π ラジアンを 100 点として帯を表示します．これを両側円帯グラフと名づけます.

②　ここではデータの大きさ順の並べかえはしません．基準線から右側，時計まわりを男子の成績，左側，反時計まわりを女子の成績とします.

③　右側の考え方は【応用問題 6】とほぼ同じですから，その解説をみてくざたい．ただし，データの並べかえがしてありませんので，データが外側より内側の方が大きい場合その帯の外側がかけないことになりますので，帯の両側をかくことにします.

【応用問題 6】のかき方

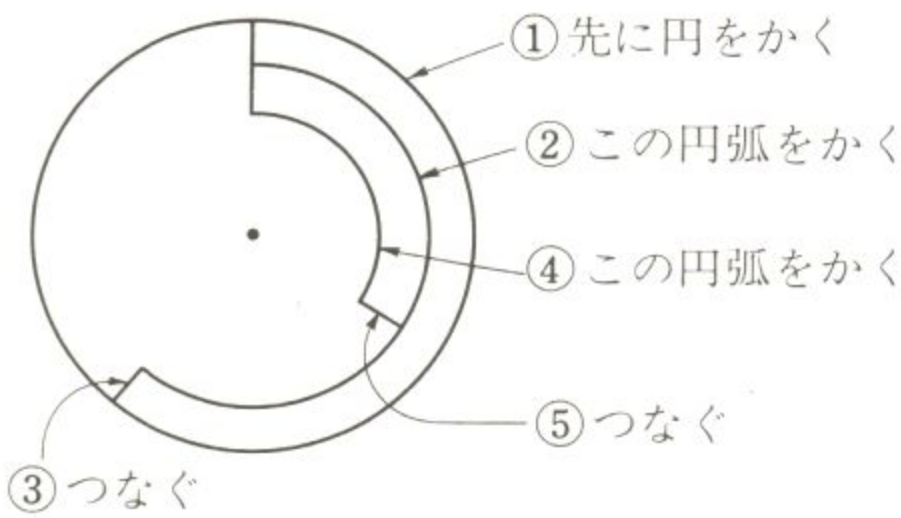
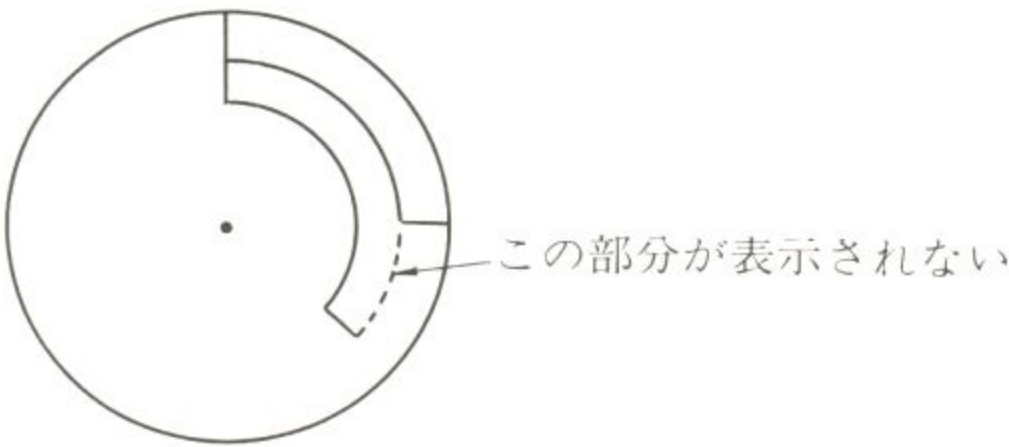
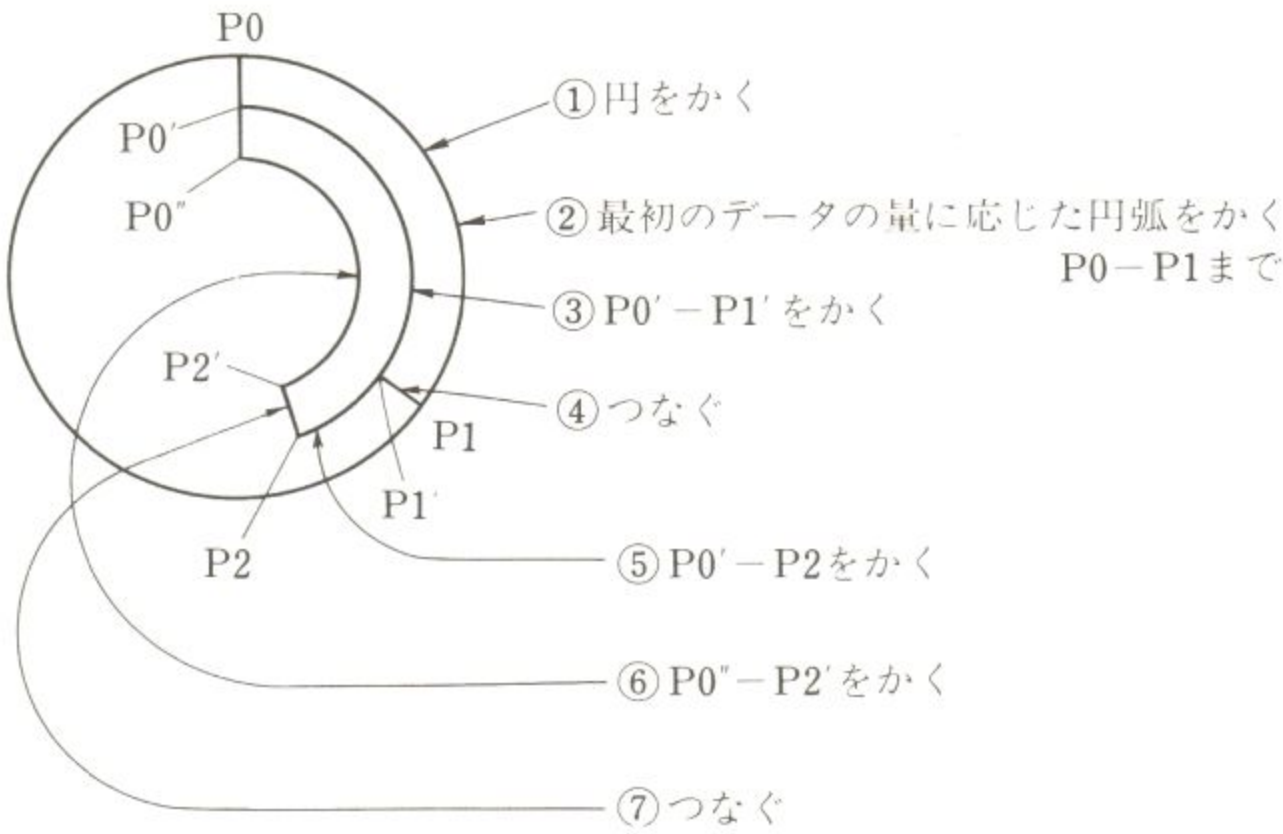
データが外側の方が大きい場合

①先に円をかく
②この円弧をかく
④この円弧をかく
⑤つなぐ
③つなぐ

データが内側の方が大きい場合

この部分が表示されない

今回のかき方

P0
P0′
P0″
P2′
P1
P1′
P2
①円をかく
②最初のデータの量に応じた円弧をかく
P0－P1まで
③P0′－P1′をかく
④つなぐ
⑤P0′－P2をかく
⑥P0″－P2′をかく
⑦つなぐ

④　左側の考えは右側と同じですが，座標値の表わし方が逆となります.

右側からかく場合の X 座標値は C+RSIN(X)，Y 座標値は C-RCOS(X)です.

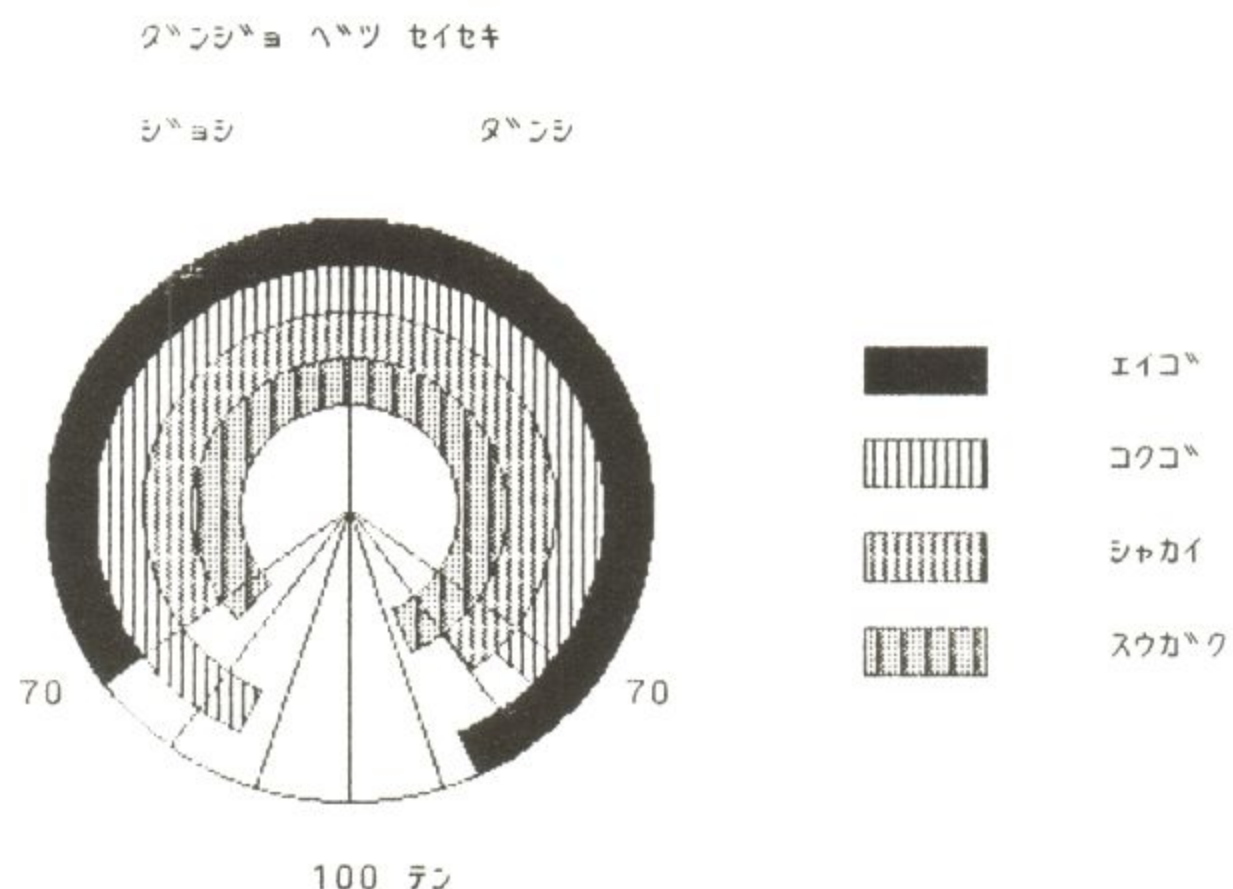
左側からかく場合はX座標値はC+RCOS(X+3.14159/2)，Y座標値はC-RSIN(X+3.14159/2)です.

【プログラム例】

```
DECLARE SUB aa ()
DECLARE SUB bb ()
DECLARE SUB cc ()
DECLARE SUB dd ()
COMMON SHARED a$(), a(), b$()
DIM a$(1 TO 6), a(1 TO 4, 1 TO 2), b$(1 TO 4)
SCREEN 0
CLS 0
DEF fnx (c, r, x) = c + r * SIN(x) ―――――――――― 円の右側の点のX座標値
DEF fny (c, r, x) = c - r * COS(x) ―――――――――― 円の右側の点のY座標値
DEF fnd (d) = d / 100 * 3.14159 ――――――――――― 成績を示す角度
DEF fna (c, r, x) = c + r * COS(x + 3.14159 / 2)―― 円の左側の点のX座標値
DEF fnb (c, r, x) = c - r * SIN(x + 3.14159 / 2)―― 円の右側の点のY座標値
b$(1) = CHR$(&HFF) ――――――――――――――――――┐ハッチング
b$(2) = CHR$(&H11)                             │
b$(3) = CHR$(&H22) + CHR$(&H11)                │
b$(4) = CHR$(&HC) + CHR$(&HAA) ―――――――――――――┘
FOR i = 1 TO 6 ―――――――――――――――――――――――┐データの読み込み，課目
    READ a$(i)                                 │
NEXT i ――――――――――――――――――――――――――――――┘
FOR i = 1 TO 4 ―――――――――――――――――――――――┐データの読み込み，点数
    READ a(i, 1), a(i, 2)                      │a(i,1)が男子，a(i,2)が女子
NEXT i ――――――――――――――――――――――――――――――┘
CALL aa ―――――――――――――――――――――――――――― グラフ準備ルーチンへ分岐
CALL bb ―――――――――――――――――――――――――――― 男子の表示
CALL cc ―――――――――――――――――――――――――――― 女子の表示
CALL dd ―――――――――――――――――――――――――――― 目盛り線の表示
END
DATA ｴｲｺﾞ,ｺｸｺﾞ,ｼｬｶｲ,ｽｳｶﾞｸ,ﾀﾞﾝｼ,ｼﾞｮｼ
DATA 85,70,75,85,80,70,88,75

SUB aa ―――――――――――――――――――――――――――――― グラフ準備ルーチン
    LOCATE 1, 22: PRINT "ﾀﾞﾝｼﾞｮ ﾍﾞﾂ ｾｲｾｷ"――――――― 標題
    FOR i = 128 TO 224 STEP 32 ―――――――――――――┐凡例
        LINE (400, i)-(440, i + 16), , B           │
        PAINT (420, i + 8), b$((i - 96) / 32)      │
    NEXT i ――――――――――――――――――――――――――――――┘
    FOR i = 1 TO 4 ―――――――――――――――――――――――┐凡例の課目名
        LOCATE i * 2 + 7, 60: PRINT a$(i)          │
    NEXT i ――――――――――――――――――――――――――――――┘
    LOCATE 19, 28: PRINT "100 ﾃﾝ" ――――――――――――┐単位，項目名
    LOCATE 3, 20: PRINT "ｼﾞｮｼ              ﾀﾞﾝｼ"――┘
END SUB

SUB bb ―――――――――――――――――――――――――――――― グラフ作成①
    CIRCLE (230, 184), 100 ―――――――――――――――― 外周の円
    LINE (230, 84)-(230, 284) ――――――――――――――― 基準線
```

```
FOR i = 1 TO 4 ―――――――――――――― 男子の 4 課目の点を示す帯
    PSET (fnx(230, 100 - (i - 1) * 16, 0), fny(184, 100 - (i - 1) * 16, 0))
        FOR q = 0 TO fnd(a(i, 1)) + .03 STEP .05 ―― 4 科目の外側の円弧
            x1 = fnx(230, 100 - (i - 1) * 16, q)
            y1 = fny(184, 100 - (i - 1) * 16, q)
            LINE -(x1, y1)
            xx1 = x1: yy1 = y1
        NEXT q
    PSET (fnx(230, 100 - i * 16, 0), fny(184, 100 - i * 16, 0)) ―― 4 科目の内側の円弧
        FOR q = 0 TO fnd(a(i, 1)) + .03 STEP .05
            x2 = fnx(230, 100 - i * 16, q)
            y2 = fny(184, 100 - i * 16, q)
            LINE -(x2, y2)
            xx2 = x2: yy2 = y2
        NEXT q
        LINE (xx1, yy1)-(xx2, yy2) ―――――――――――― 帯の終端の線
        x3 = fnx(230, 100 - (i - 1) * 16 - 8, fnd(a(i, 1) / 2)) ―― ハッチング
        y3 = fny(184, 100 - (i - 1) * 16 - 8, fnd(a(i, 1) / 2))
        PAINT (x3, y3), b$(i)
    NEXT i
END SUB

SUB cc ―――――――――――――――――――――― グラフ作成②
    FOR i = 1 TO 4 ―――――――――――――――― 女子の 4 課目の点を示す帯
        PSET (fna(230, 100 - (i - 1) * 16, 0), fnb(184, 100 - (i - 1) * 16, 0))
        FOR q = 0 TO fnd(a(i, 2)) + .03 STEP .05 ――― 4 課目の外側の円弧
            x1 = fna(230, 100 - (i - 1) * 16, q)
            y1 = fnb(184, 100 - (i - 1) * 16, q)
            LINE -(x1, y1)
            xx1 = x1: yy1 = y1
        NEXT q
        PSET (fna(230, 100 - i * 16, 0), fnb(184, 100 - i * 16, 0)) ―― 4 課目の内側の円弧
        FOR q = 0 TO fnd(a(i, 2)) + .03 STEP .05
            x2 = fna(230, 100 - i * 16, q)
            y2 = fnb(184, 100 - i * 16, q)
            LINE -(x2, y2)
            xx2 = x2: yy2 = y2
        NEXT q
        LINE (xx1, yy1)-(xx2, yy2) ―――――――――――― 帯の終端の線
        x3 = fna(230, 100 - (i - 1) * 16 - 8, fnd(a(i, 2) / 2)) ―― ハッチング
        y3 = fnb(184, 100 - (i - 1) * 16 - 8, fnd(a(i, 2) / 2))
        PAINT (x3, y3), b$(i)
    NEXT i
END SUB

SUB dd ――――――――――――――――――――――― 目盛り線と目盛り
    FOR i = 70 TO 90 STEP 10 ――――――――――――― 男子側の目盛り線
        px = fnx(230, 100, fnd(i))
        py = fnb(184, 100, fnd(i))
        LINE (230, 184)-(px, py)
    NEXT i
```

```
    FOR i = 70 TO 90 STEP 10 ――――――――― 女子側の目盛り線
        px = fna(230, 100, fnd(i))
        py = fnb(184, 100, fnd(i))
        LINE (230, 184)-(px, py)
    NEXT i
    LOCATE 16, 16: PRINT "70" ―――――――― 目盛り数値
    LOCATE 16, 42: PRINT "70" ―――――――― 目盛り数値
END SUB
```

【結　果】

【応用問題8】　虹グラフ

次表を虹グラフとするプログラムをつくれ.

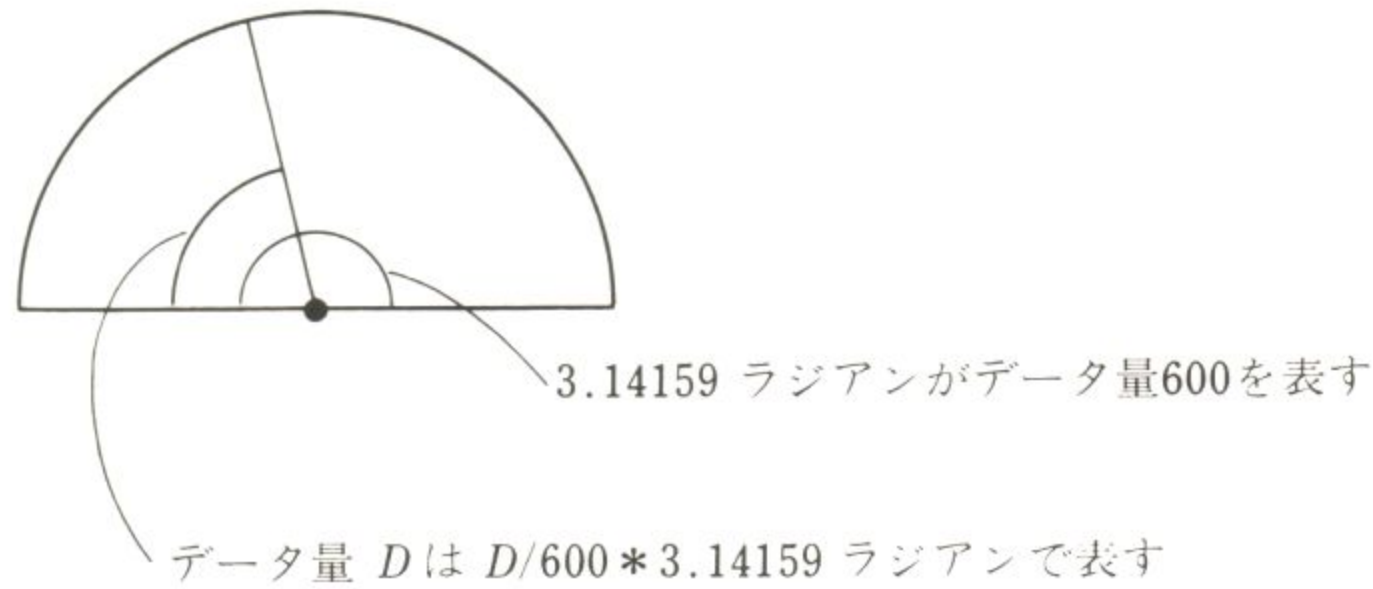
商品別売上高　　（1988年）

商　品	売上高（万円）
クーラー	250
モーター	220
ＶＴＲ	540
ステレオ	320
カメラ	100

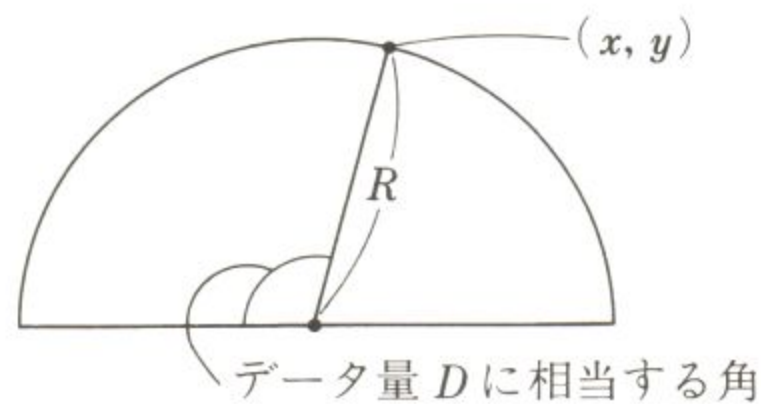
【解　説】　◎　虹グラフをつくります.

① 結果のように帯グラフの応用で左端を基準として，半円弧の右端までをデータ量の100%とする帯を外から内へ何重かに表わしたものを虹グラフと名づけます.

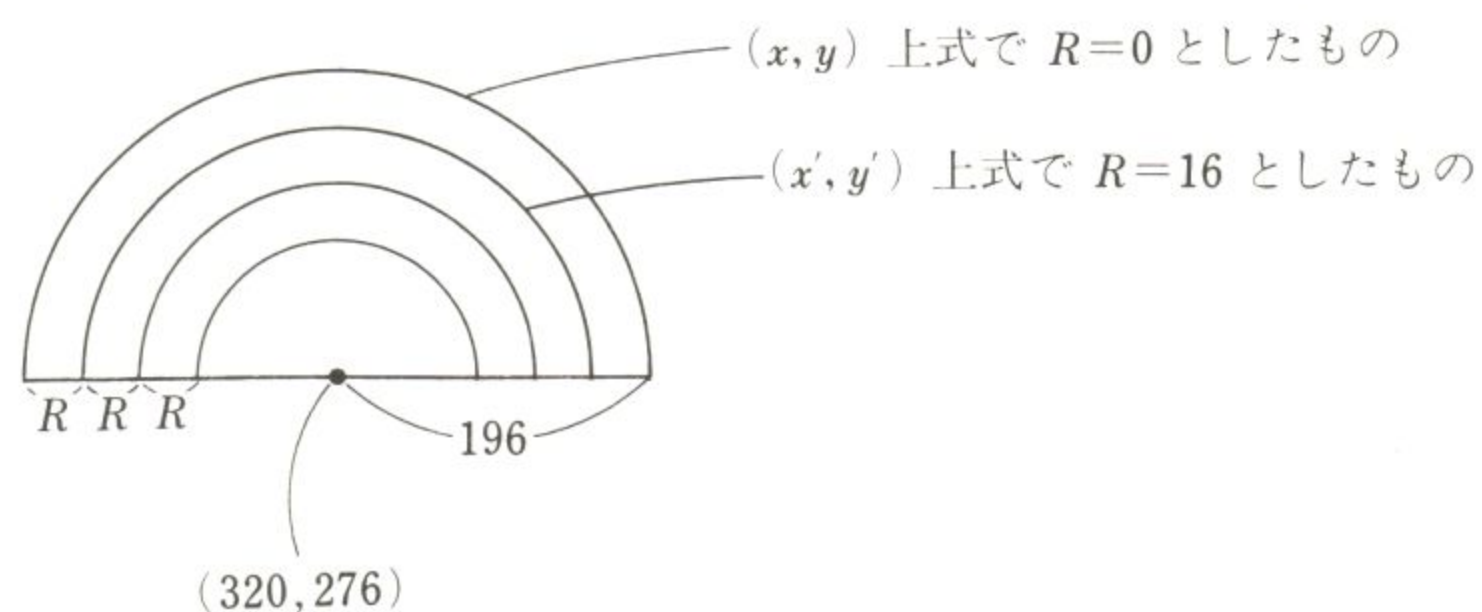
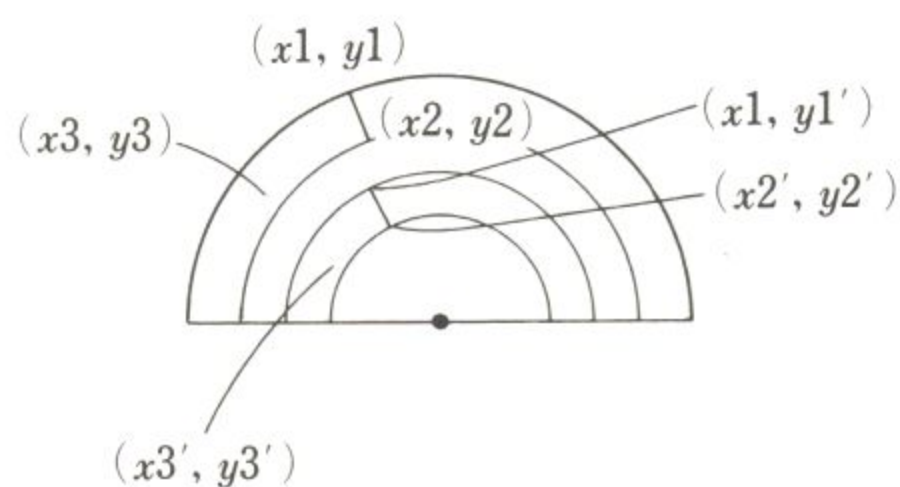
② 考え方の基本は【応用問題6】以降の帯グラフと同じです．基準が【応用問題6】では12時，データ量の100%を全周で表わしているのに対し，本グラフでは基準を9時の位置，データ量の100%を時計方向半周で表わします.

③ この場合のデータ量の最大を600とすると，データ量のDの角度はD/600*3.14159ラジアンとなります.

④　データ量Dの角をXとするとX，Y座標は半径をRとすれば次のようになります．

X＝R*SIN(X+3.14159/2)

Y＝R*COS(X+3.14159/2)

⑤　中心を(320,276)とし，一番外側の半円の半径を196とし，内側の半円との間隔をRで与えなおすと

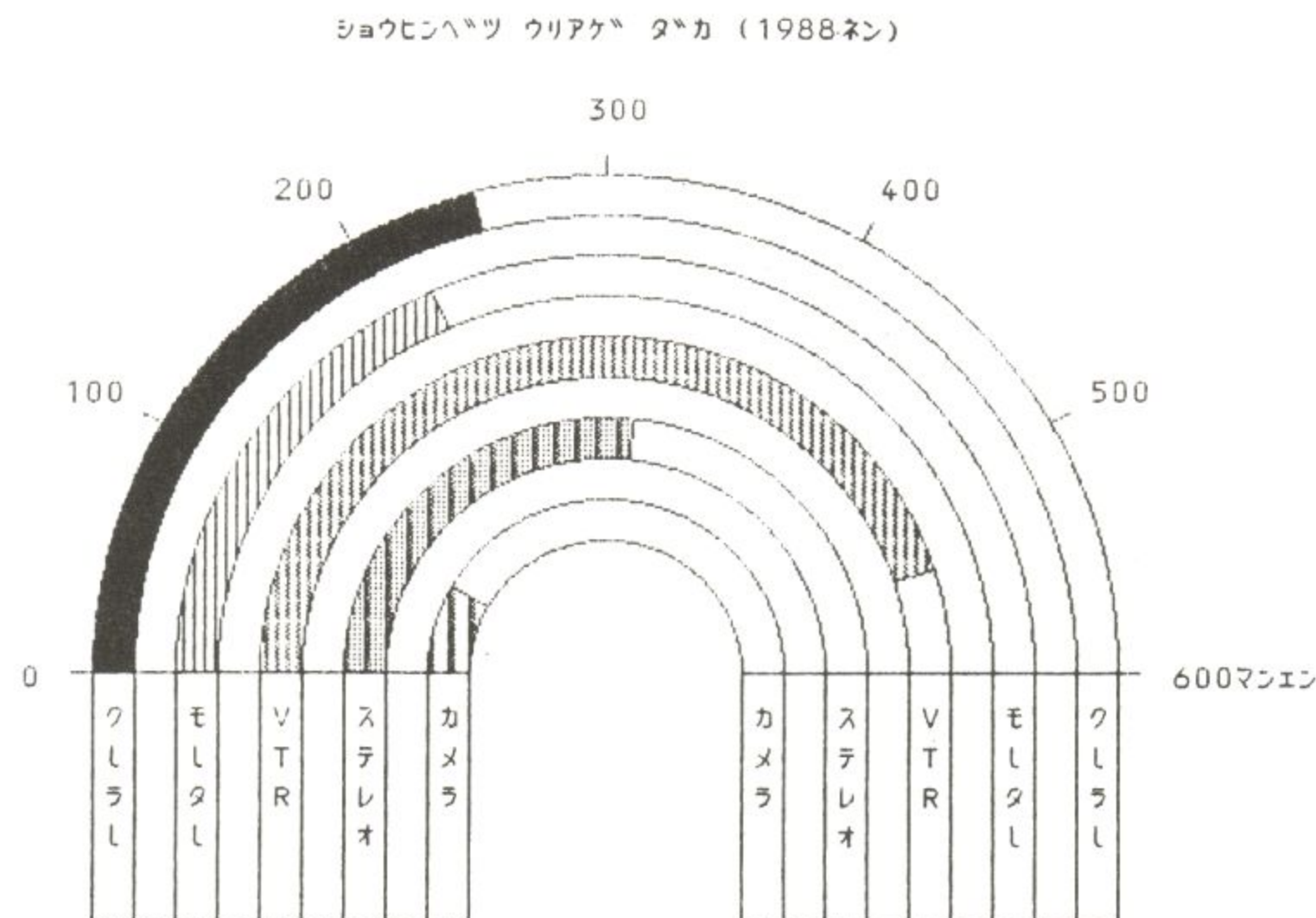
X=320-(196+R)*SIN(X+3.14159/2)

Y=276+(196+R)*COS(X+3.14159/2)となります．

⑥　具体的には次のような手順で表示します．

① クーラーの値250に相当する角の点　X1，Y1を求めます．半径は196です．
② クーラーの値250に相当する角の点　X2，Y2を求めます．半径は196-16です．
③ (X1,Y1)と(X2,Y2)を結びます．
④ 中間点(X3,Y3)を求めます．
⑤ (X3,Y3)を指定してPAINTします．
⑥ ①〜⑤をモーターの値220について行います．①に相当するRは16*2で，半径は196-32です．②に相当するRは16*3で48，半径は196-48です．
⑦ 以下同様にVTR，ステレオ，カメラについて行います．

【プログラム例】

```
DECLARE SUB aa ()
DECLARE SUB bb ()
COMMON SHARED a$(), a(), b$()
DIM a$(1 TO 5), a(1 TO 5), b$(1 TO 5)
SCREEN 0
CLS 0
DEF fnx (r, x) = 320 - (196 + r) * SIN(x + 3.14159 / 2)――┐データ量に相当する半円周上の
DEF fny (r, y) = 276 + (196 + r) * COS(y + 3.14159 / 2)――┘点の x, y 座標値
DEF fnd (d) = d / 600 * 3.14159 ――――――――――― データ量に相当する角度
b$(1) = CHR$(&HFF) ―――――――――――――――― ハッチングパターンの登録
b$(2) = CHR$(&H11)
b$(3) = CHR$(&H22) + CHR$(&H11)
b$(4) = CHR$(&HC) + CHR$(&HAA)
b$(5) = CHR$(&HC) + CHR$(&HA) ―――――――
FOR i = 1 TO 5 ―――――――――――――――――――― データの読み込み
   READ a$(i), a(i)
NEXT i ―――――――――――――――――――――――
CALL aa ―――――――――――――――――――――― グラフ準備ルーチンへ分岐
CALL bb ―――――――――――――――――――――― グラフ作成ルーチンへ分岐
END
DATA ｸﾘｰﾗｰ ,250,ﾓｰﾀｰ ,220,VTR  ,540
DATA ｽﾃﾚｵ ,320,ｶﾒﾗ ,100

SUB aa ――――――――――――――――――――――― グラフ準備
    LOCATE 1, 28: PRINT "ｼｮｳﾋﾝﾍﾞﾂ ｳﾘｱｹﾞﾀﾞｶ(1987ﾈﾝ)"―――― 標題
    FOR i = 0 TO 9 ――――――――――――――――――― 直線+円弧の表示
       LINE (128 + i * 16 - 4, 276)-(128 + i * 16 - 4, 376)
       CIRCLE (320, 276), 196 - 16 * i, , 0, 3.14159
       LINE (520 - i * 16 - 4, 276)-(520 - i * 16 - 4, 376)
    NEXT i ―――――――――――――――――――――――
    LINE (116, 276)-(268, 276) ―――――――――――――― 部品名をかく部分の横線
    LINE (524, 276)-(372, 276)
    LINE (124, 376)-(268, 376)
    LINE (516, 376)-(372, 376) ――――――――――――――
    FOR i = 1 TO 5 ―――――――――――――――――――― 商品名
       LOCATE 19, 13 + 4 * i: PRINT MID$(a$(i), 1, 1)
       LOCATE 20, 13 + 4 * i: PRINT MID$(a$(i), 2, 1)
       LOCATE 21, 13 + 4 * i: PRINT MID$(a$(i), 3, 1)
       LOCATE 22, 13 + 4 * i: PRINT MID$(a$(i), 4, 1)
       LOCATE 19, 44 + 4 * i: PRINT MID$(a$(6 - i), 1, 1)
       LOCATE 20, 44 + 4 * i: PRINT MID$(a$(6 - i), 2, 1)
       LOCATE 21, 44 + 4 * i: PRINT MID$(a$(6 - i), 3, 1)
       LOCATE 22, 44 + 4 * i: PRINT MID$(a$(6 - i), 4, 1)――――
       LOCATE 18, 13: PRINT "0" ―――――――――――――― 目盛り数値と単位
       LOCATE 11, 15: PRINT "100"
       LOCATE 6, 25: PRINT "200"
       LOCATE 4, 40: PRINT "300"
       LOCATE 6, 54: PRINT "400"
       LOCATE 11, 64: PRINT "500"
       LOCATE 18, 68: PRINT "600ﾏﾝｴﾝ" ――――――――――――

       ii = i * 100
       LINE (fnx(8, fnd(ii)), fny(8, fnd(ii)))-(fnx(0, fnd(ii)), fny(0, fnd(ii)))― 目盛り線
    NEXT i
END SUB
SUB bb ――――――――――――――――――――― 帯表示
```

```
    FOR i = 1 TO 5 ―――――――――――――――帯の終端部の表示
        ii = (4 - 4 * i) * 8
        iii = ii - 16                                                          区切り線
        LINE (fnx(ii, fnd(a(i))), fny(ii, fnd(a(i))))-(fnx(iii, fnd(a(i))), fny(iii, fnd(a(i)))
        PAINT (fnx((ii - 4), fnd(a(i) / 2)), fny((ii - 4), fnd(a(i) / 2))), b$(i)――ハッチング
    NEXT i
END SUB
```

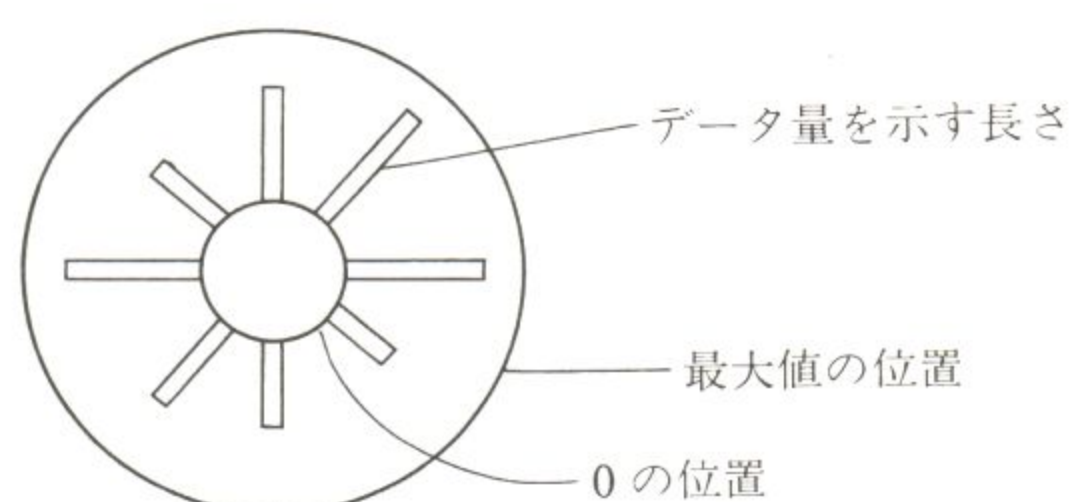
【結　果】

【応用問題9】　放射線グラフ

下の表を放射線グラフにするプログラムをつくれ.

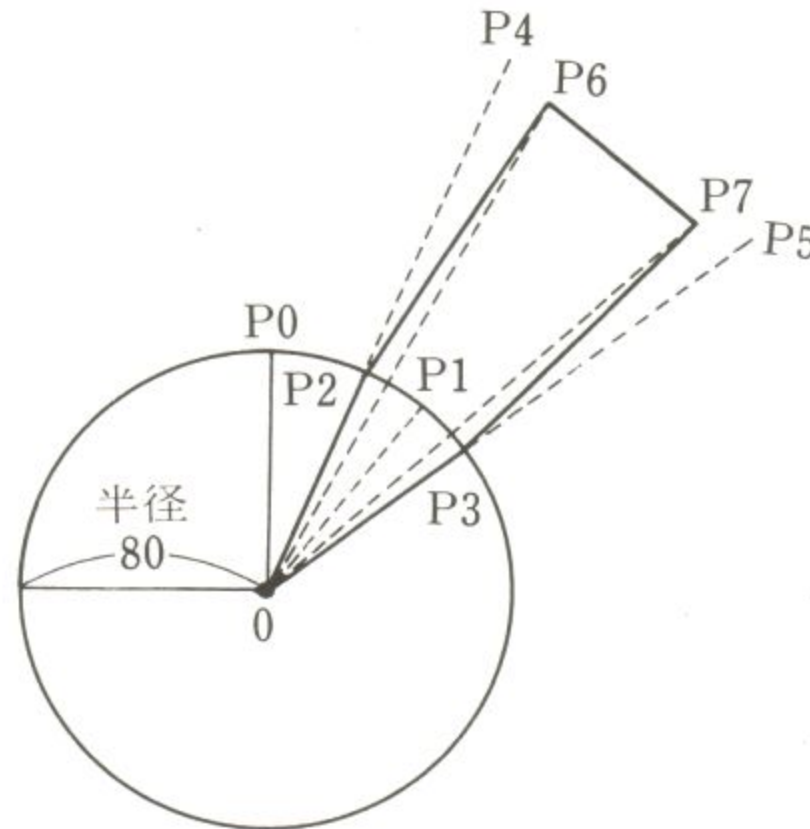
交通量調査（1989年10月10日）

時間	1000台	時間	1000台	時間	1000台	時間	1000台
1	100	7	250	13	180	19	490
2	50	8	500	14	190	20	420
3	60	9	400	15	250	21	320
4	90	10	350	16	400	22	250
5	120	11	300	17	470	23	200
6	180	12	250	18	500	24	100

【解　説】　◎　放射グラフをつくります.

①　中心からデータ量に応じた長さの棒を放射線状にかくグラフを放射線グラフと名づけます.

② 中心を(300,200)とし，半径を80の円をかきます．これが真中の円で統計量0の基準となります．

③ ここでは1時から24時まで24個の時系列データを時計まわりで表示します．

④ 1時のデータを100とすると次のような手順で表示します．

① 全体で24本の棒を表示しますから，1時の位置角P0-0-P1は1/24*2*3.14159ラジアンです．P0-0-P1の角を中心にして棒の幅をもたせるため-0.8ラジアンと+0.8ラジアンの角を考えます．中心(300,200)，半径80で，角を1/24*2*3.14159-0.8ラジアンの点がP2，角を1/24*2*3.14159+0.8ラジアンの点がP3です．P2が棒のかき始めの点，P3が棒のかきおえた点です．
② 次に棒の長さは半径20ドットについて100台としますので，データ量/100*20が棒の長さです．中心からの長さ（半径）ではそれに80を加えたものです．
③ したがって，中心を（300,200），半径を80+100/20角度を1/24*2*3.14159-0.8ラジアンの点P4と，角度を1/24*2*3.14159+0.8ラジアンの点P5を求めP2-P4-P5-P3と結ぶと棒になります．しかし，棒の長さが長くなればなるほど，先が開く扇形になりますので，一工夫します．
④ P0-0-P1から-0.6ラジアン，+0.6ラジアンの角を先端では使います．半径（300,200），半径80+100/100*20角度を1/24*2*3.14159-0.6ラジアンの点をP6，角度を1/24*2*3.14159+0.6ラジアンの点をP7とします．P2-P6-P7-P3を結びます．棒となります．

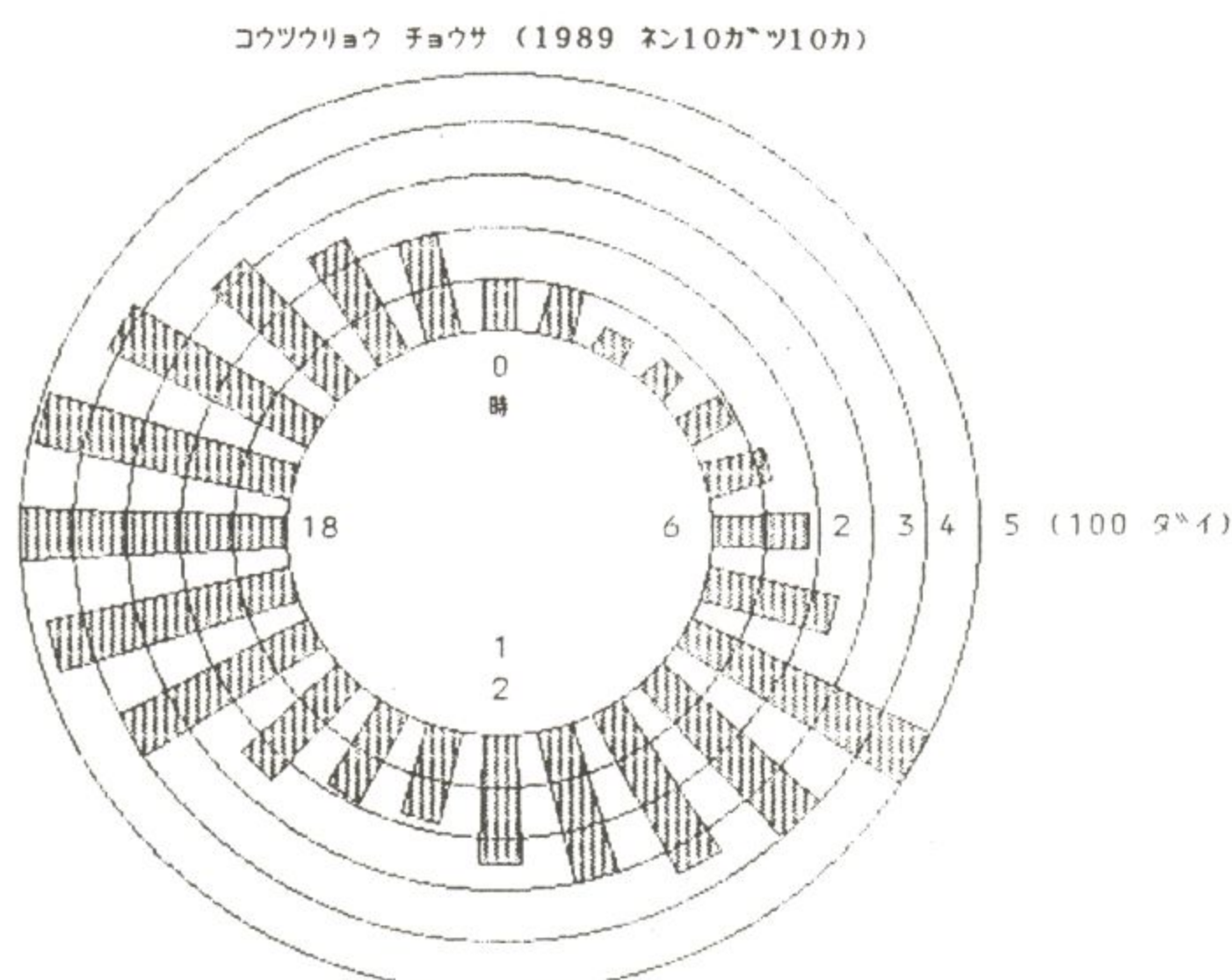
【プログラム例】

```
DECLARE SUB aa ()
DECLARE SUB bb ()
COMMON SHARED a(), b$
DIM a(1 TO 24)
SCREEN 0
CLS 0
DEF fnx (c, r, x) = c + r * SIN(x) ―― 中心のx座標をc，半径をr，角をxとする円周上の点のx座標値
DEF fny (c, r, x) = c - r * COS(x) ―― 中心のY座標値をc，半径をr，角をxとする円周上の文のY座標値
DEF fnd (d, e) = d / 24 * 2 * 3.14159 + e ―― D時（Dは1から24）のときの角を中心として+e，-eラジアンの角
DEF fne (f) = f / 100 * 20 + 80 ―― データ量Fに相当する棒の長さ+80でFに相当する半径を示す
b$ = CHR$(&H22) + CHR$(&H11) ―― ハッチング
FOR i = 1 TO 24 ―― データの読み込み
    READ a(i)
NEXT i
CALL aa ―― グラフ準備ルーチンへ分岐
CALL bb ―― グラフ作成ルーチンへ分岐
END
DATA 100,50,60,90,120,180,250,500,400,350,300,250
DATA 180,190,250,400,470,500,490,420,320,250,200,100
```

```
SUB aa ――――――――――――――――――――――――――――――― グラフ準備ルーチン
    LOCATE 1, 26: PRINT "ｺｳﾂｳﾘｮｳ ﾁｮｳｻ (1989 ﾈﾝ10ｶﾞﾂ10ｶ)" ―― 標題
    LOCATE 9, 37: PRINT 0 ――――――――――――――――― 時間と目盛り数値，単位
    LOCATE 10, 38: PRINT "ｼﾞ"
    LOCATE 13, 45: PRINT 6
    LOCATE 16, 37: PRINT 1
    LOCATE 17, 37: PRINT 2
    LOCATE 13, 28: PRINT 18
    LOCATE 13, 53: PRINT 2
    LOCATE 13, 56: PRINT 3
    LOCATE 13, 58: PRINT 4
    LOCATE 13, 62: PRINT "5 (100ﾀﾞｲ)"
END SUB

SUB bb ―――――――――――――――――――――――― グラフ作成ルーチン
    CIRCLE (300, 200), 80 ―――――――――――― 円
    FOR i = 1 TO 24 ――――――――――――――――― 棒の作成
        PSET (fnx(300, fne(0), fnd(i, -.08)), fny(200, fne(0), fnd(i, -.08))) ―― 棒
        x1 = fnx(300, fne(a(i)), fnd(i, -.06))
        y1 = fny(200, fne(a(i)), fnd(i, -.06))
        x2 = fnx(300, fne(a(i)), fnd(i, .06))
        y2 = fny(200, fne(a(i)), fnd(i, .06))
        x3 = fnx(300, fne(0), fnd(i, .08))
        y3 = fny(200, fne(0), fnd(i, .08))
        LINE -(x1, y1)
        LINE -(x2, y2)
        LINE -(x3, y3)
        x4 = (x1 + x3) / 2 ――――――――――― ハッチング
        y4 = (y1 + y3) / 2
        PAINT (x4, y4), b$
     NEXT i
     FOR i = 100 TO 180 STEP 20 ―――――――― 同心円
         CIRCLE (300, 200), i
     NEXT i
END SUB
```

【結　果】

ｺｳﾂｳﾘｮｳ ﾁｮｳｻ (1989 ﾈﾝ10ｶﾞﾂ10ｶ)

0
時
18
6
2 3 4 5 (100 ﾀﾞｲ)
1
2

【応用問題 10】 レーダーチャート

下表をレーダーチャートにかくプログラムをつくれ.

月別売上

月	売上（100 万円）	月	売上（100 万円）
1	80	7	140
2	60	8	90
3	180	9	150
4	160	10	160
5	170	11	130
6	120	12	120

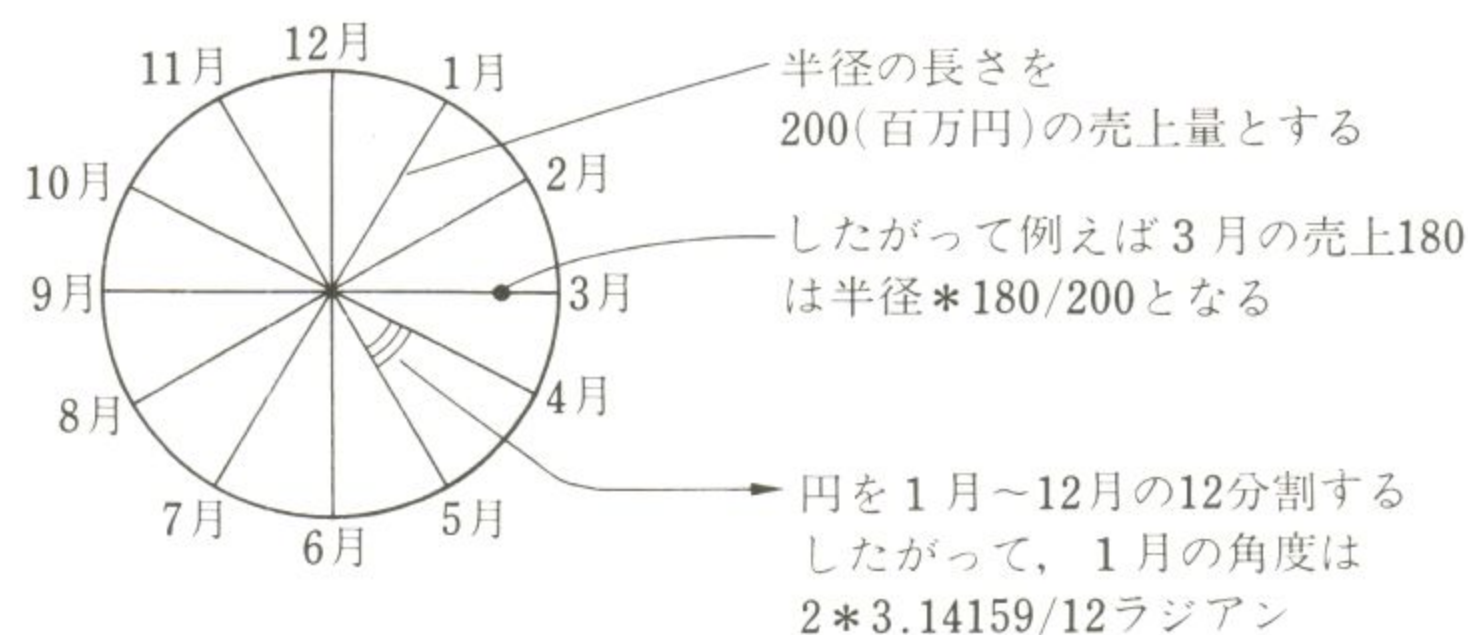
【解 説】 ◎ レーダーチャートをつくります.

① 下図のような考えでレーダーチャートをつくります.

12月 1月 2月 3月 4月 5月 6月 7月 8月 9月 10月 11月

半径の長さを
200(百万円)の売上量とする

したがって例えば3月の売上180
は半径＊180/200となる

円を1月～12月の12分割する
したがって, 1月の角度は
2＊3.14159/12ラジアン

したがって, 中心をC, データ量をD, 円の半径を200とするとM円の値のドットの表示位置は
X=C+200*180/200*SIN(M*2*3.14159/12)
Y=C−200*180/200*COS(M*2*3.14159/12)となります.

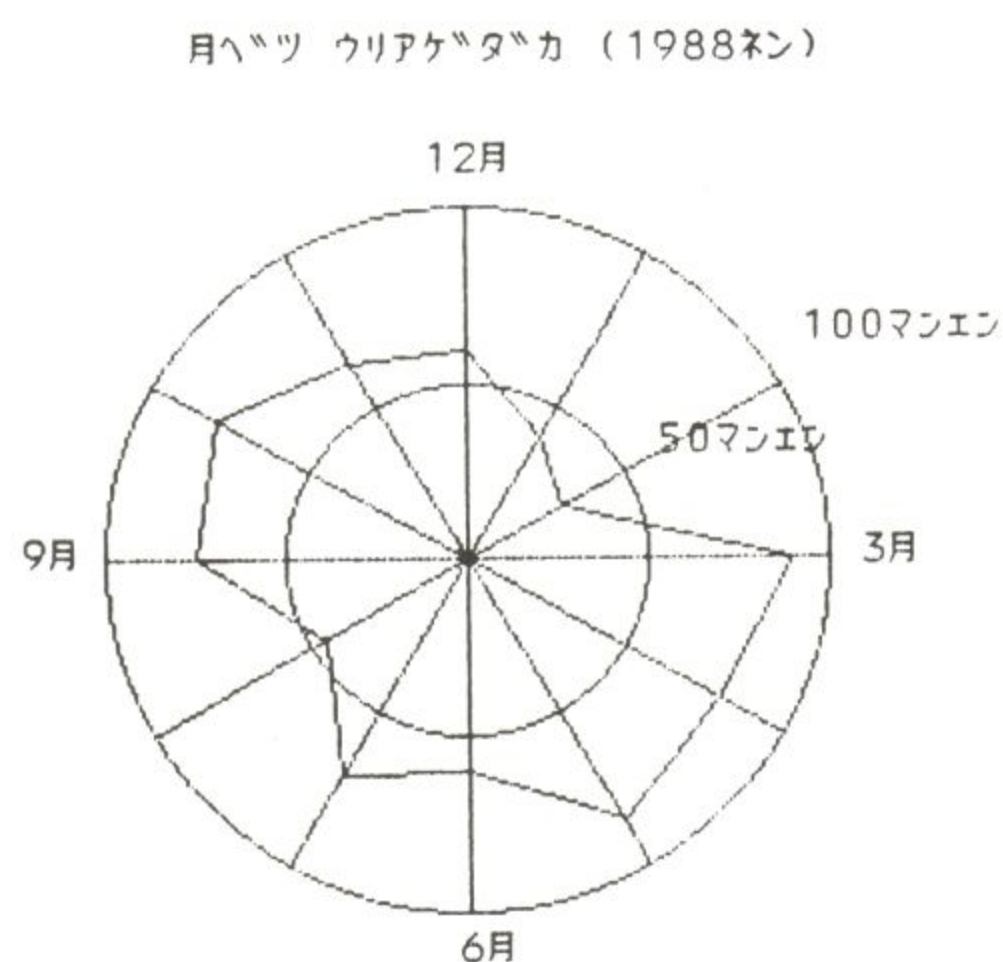
【プログラム例】

```
DECLARE SUB aa ()
DECLARE SUB bb ()
COMMON SHARED a()
DIM a(1 TO 12)
SCREEN 0
CLS 0
DEF fnx (c, r, x) = c + r * SIN(x / 12 * 2 * 3.14159)  ─┬─ データ量をプロット
DEF fny (c, r, x) = c - r * COS(x / 12 * 2 * 3.14159)  ─┘  X, Y座標値
DEF fnd (d) = d / 200 * 100 ──────────────────── データ量に相当する半径
FOR i = 1 TO 12 ──────┬─ データの読み込み          円の半径が100
    READ a(i) ────────┘
NEXT i
CALL aa ──────────────── グラフ作成準備ルーチンへ分岐
CALL bb ──────────────── グラフ作成ルーチンへ分岐
END
DATA 80,60,180,160,170,120
DATA 140,90,150,160,130,120
```

```
SUB aa ―――――――――――――――――――――――――――――― グラフ作成準備ルーチン
    LOCATE 4, 29: PRINT "ﾂｷﾍﾞﾂ ｳﾘｱｹﾞﾀﾞｶ (1988 ﾈﾝ)"―― 標題
    LOCATE 6, 37: PRINT "12ｶﾞﾂ"――――――――――――――┬ 項目名, 単位
    LOCATE 13, 52: PRINT "3ｶﾞﾂ"                   │
    LOCATE 20, 38: PRINT "6ｶﾞﾂ"                   │
    LOCATE 13, 21: PRINT "9ｶﾞﾂ"                   │
    LOCATE 9, 50: PRINT "100ﾏﾝｴﾝ"                 │
    LOCATE 11, 45: PRINT "50ﾏﾝｴﾝ"―――――――――――――┘
    CIRCLE (300, 200), 100 ―――――――――――――――――――┬ 円と目盛り線の表示
    CIRCLE (300, 200), 50                          │
    FOR i = 1 TO 12                                │
        x = fnx(300, 100, i)                       │
        y = fny(200, 100, i)                       │
        LINE (300, 200)-(x, y)                     │
    NEXT i ―――――――――――――――――――――――――――――――――――┘
END SUB

SUB bb ――――――――――――――――――――――――――――――――――――――― グラフ作成ルーチン
    PSET (fnx(300, fnd(a(12)), 0), fny(200, fnd(a(12)), 0))─┬ プロット
    FOR i = 1 TO 12                                          │ 12 月-1 月
        x = fnx(300, fnd(a(i)), i)                           │ -2 月……
        y = fny(200, fnd(a(i)), i)                           │ -12 月のデータを順にむすぶ
        LINE -(x, y)                                         │
    NEXT i ――――――――――――――――――――――――――――――――――――――――――――――┘
END SUB
```

【結　果】

ﾂｷﾍﾞﾂ ｳﾘｱｹﾞﾀﾞｶ (1988ﾈﾝ)

12ｶﾞﾂ　100ﾏﾝｴﾝ　50ﾏﾝｴﾝ　3ｶﾞﾂ　9ｶﾞﾂ　6ｶﾞﾂ

【応用問題11】 絵画グラフ

下の業種別景気状況を，景気の状態ごとの顔色で表わす絵画グラフをつくれ．なお，景気の状況は①超繁忙 ②好景気 ③普通 ④不調 ⑤不景気とし，おのおのの顔色は，結果のようなマークとする．

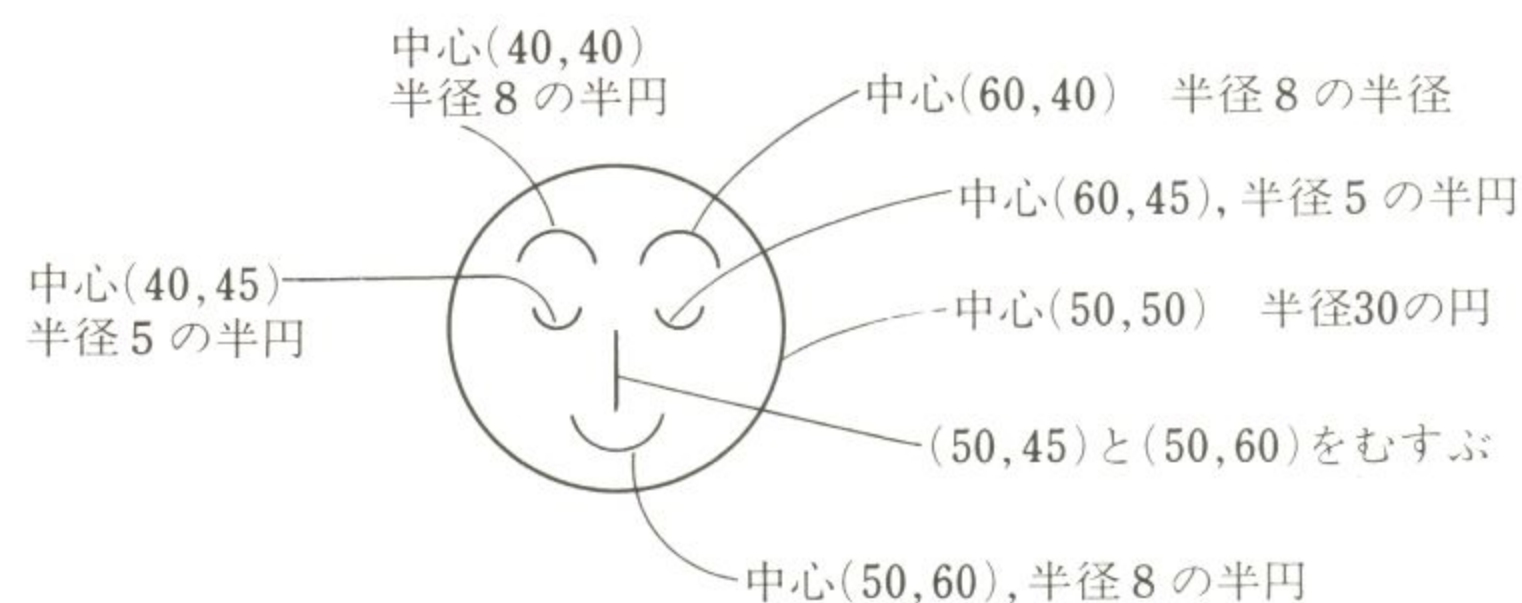
業種別景気予想

業　種	景気予想
化学工業	好景気
鉄鋼業	普　通
繊維工業	不　況
電機工業	不　調
機械工業	超繁忙
商　業	不　況

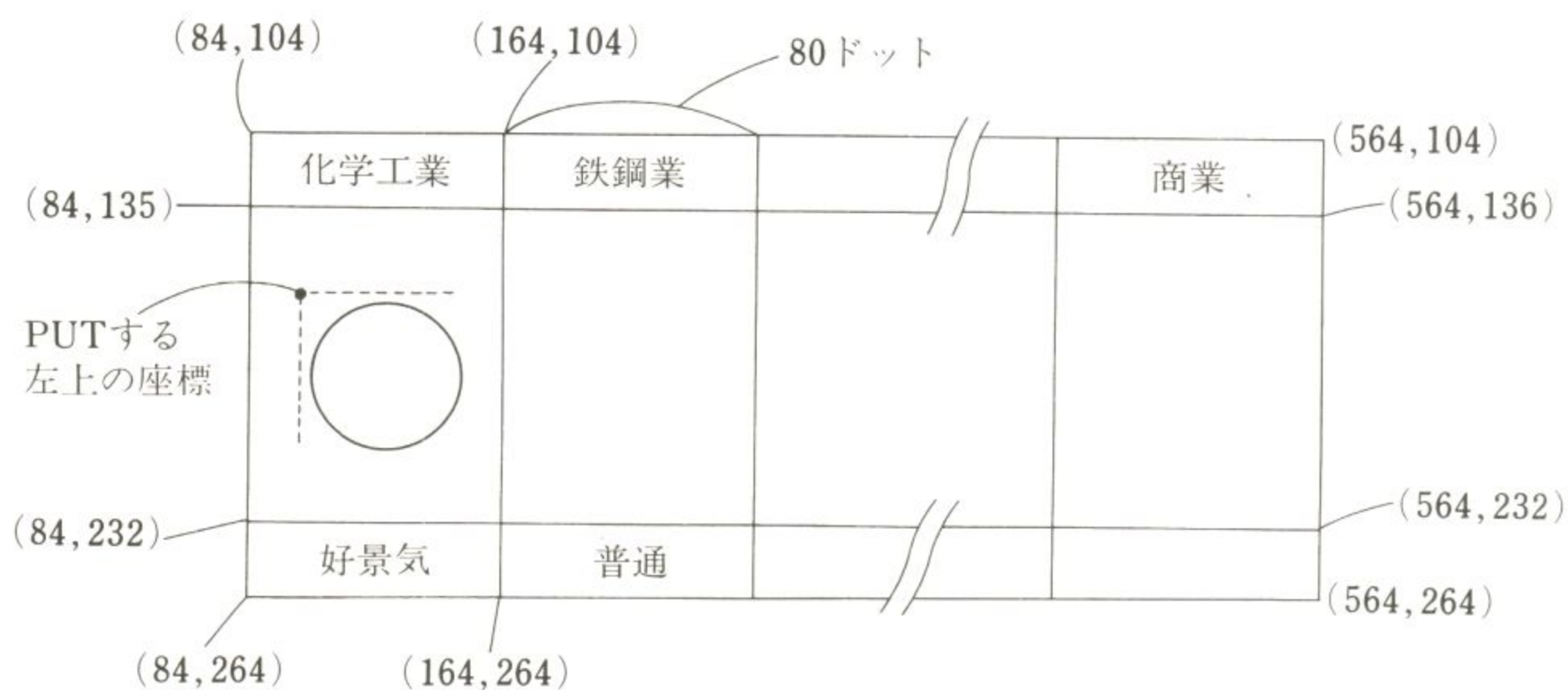
【解　説】 ◎ 絵画グラフをつくります．

① 景気の状況による顔色を5種用意します．おのおの，グラフィックで表示した後配列 A%～E%に GET@でとりこみます．

好景気の例を示します．

中心(40,40)
半径8の半円
中心(60,40) 半径8の半径
中心(60,45)，半径5の半円
中心(40,45)
半径5の半円
中心(50,50) 半径30の円
(50,45)と(50,60)をむすぶ
中心(50,60)，半径8の半円

左上 (20,20) と右下 (80,80) を指定して配列 B%に取り込む

② 表の枠，文字を表示し，絵画の部分に顔色が格納されている配列を PUT で表示します．

(84,104)
(164,104)
80ドット
(564,104)
化学工業
鉄鋼業
商業
(84,135)
(564,136)
PUTする
左上の座標
(84,232)
(564,232)
好景気
普通
(564,264)
(84,264)
(164,264)

【プログラム例】

```
DECLARE SUB aa ()
DECLARE SUB bb ()
DECLARE SUB cc ()
DECLARE SUB dd ()
DECLARE SUB ee ()
DECLARE SUB ff ()
DECLARE SUB gg ()
COMMON SHARED a%(), b%(), c%(), d%(), e%(), a$(), b$()  ――― 5種の顔の入る配列
DIM a%(1000), b%(1000), c%(1000), d%(1000), e%(1000)
DIM a$(1 TO 6, 1 TO 2), b$(1 TO 5)
SCREEN 0
CLS 0
FOR i = 1 TO 6  ――― a$(i,1)に業種名, a$(i,2)に景気の状態1～5を読み込む
   READ a$(i, 1), a$(i, 2)
NEXT i
FOR i = 1 TO 5  ――― 景気状態の文字列を読み込む
   READ b$(i)
NEXT i
CALL aa  ――― 景気ごとの顔のパターンの取り込み
CALL bb
CALL cc
CALL dd
CALL ee
CLS 0
CALL ff  ――― グラフ作成準備ルーチンへ分岐
CALL gg  ――― グラフ作成ルーチン
END
DATA 化学工業,2,鉄鋼業,3,繊維工業,5
DATA 電機工業,4,機械工業,1,商業,5
DATA 超繁忙,好景気,普通,不調,不況

SUB aa  ――― 顔色1の配列a%への取り込み
   CLS 0
   CIRCLE (50, 50), 30
   CIRCLE (40, 40), 8, , 0, 3.14159
   CIRCLE (60, 40), 8, , 0, 3.14159
   CIRCLE (40, 45), 5, , 0, 3.14159
   CIRCLE (60, 45), 5, , 0, 3.14159
   LINE (50, 45)-(50, 60)
   CIRCLE (50, 60), 8, , 3.14159, 2 * 3.14159
   GET (20, 20)-(80, 80), a%
END SUB

SUB bb  ――― 顔色2の配列b%への取り込み
   CLS 0
   CIRCLE (50, 50), 30
   CIRCLE (40, 40), 8, , 0, 3.14159
   CIRCLE (60, 40), 8, , 0, 3.14159
   CIRCLE (40, 45), 5, , 3.14159, 2 * 3.14159
   CIRCLE (60, 45), 5, , 3.14159, 2 * 3.14159
```

```
    LINE (50, 45)-(50, 60)
    CIRCLE (50, 60), 8, , 3.14159, 2 * 3.14159
    GET (20, 20)-(80, 80), b%
END SUB

SUB cc ―――――――――――――――――――― 顔色 3 の配列 c%への取り込み
    CLS 0
    CIRCLE (50, 50), 30
    CIRCLE (40, 40), 8, , 0, 3.14159
    CIRCLE (60, 40), 8, , 0, 3.14159
    CIRCLE (40, 45), 5, , 3.14159, 2 * 3.14159
    CIRCLE (60, 45), 5, , 3.14159, 2 * 3.14159
    LINE (50, 45)-(50, 60)
    LINE (40, 65)-(60, 65)
    GET (20, 20)-(80, 80), c%
END SUB

SUB dd ―――――――――――――――――――― 顔色 4 の配列 d%への取り込み
    CLS 0
    CIRCLE (50, 50), 30
    LINE (32, 35)-(48, 35)
    LINE (52, 35)-(68, 35)
    LINE (34, 45)-(46, 45)
    LINE (54, 45)-(66, 45)
    LINE (50, 47)-(50, 60)
    LINE (40, 65)-(60, 65)
    GET (20, 20)-(80, 80), d%
END SUB

SUB ee ―――――――――――――――――――― 顔色 5 の配列 e%への取り込み
    CLS 0
    CIRCLE (50, 50), 30
    LINE (35, 30)-(45, 40)
    LINE (55, 40)-(65, 30)
    LINE (32, 55)-(45, 45)
    LINE (55, 45)-(68, 55)
    LINE (50, 50)-(50, 60)
    LINE (40, 70)-(50, 65)
    LINE -(60, 70)
    GET (20, 20)-(80, 80), e%
END SUB

SUB ff ――――――――――――――――――――――――――――― グラフ作成準備ルーチン
    LOCATE 5, 33: PRINT "業種別景気予想" ――― 標題
    FOR i = 84 TO 564 STEP 80 ――――――――――┐ 表のわく
        LINE (i, 104)-(i, 264)                │
    NEXT i                                    │
    LINE (84, 104)-(564, 104)                 │
    LINE (84, 135)-(564, 136)                 │
    LINE (84, 232)-(564, 232)                 │
    LINE (84, 264)-(564, 264) ――――――――――――┘
END SUB
```

```
SUB gg ――――――――――――――――――――――――― 絵の表示
    FOR i = 1 TO 6 ――――――――――――――――― 6 業種名の表示
    LOCATE 8, 2 + i * 10: PRINT a$(i, 1)――― 景気状態の表示
    x = VAL(a$(i, 2))
    LOCATE 16, 4 + i * 10: PRINT b$(x)
    IF a$(i, 2) = "1" THEN PUT (16 + 80 * i, 150), a% ―┬― 顔の表示
    IF a$(i, 2) = "2" THEN PUT (16 + 80 * i, 150), b%  │
    IF a$(i, 2) = "3" THEN PUT (16 + 80 * i, 150), c%  │
    IF a$(i, 2) = "4" THEN PUT (16 + 80 * i, 150), d%  │
    IF a$(i, 2) = "5" THEN PUT (16 + 80 * i, 150), e% ―┘
NEXT i
END SUB
```

【結　果】

業種別景気予想

化学工業	鉄鋼業	繊維工業	電機工業	機械工業	商業
好景気	普通	不況	不調	超繁忙	不況

4章　媒介変数を使った図形

【応用問題 12】　円

円を表示するプログラムをつくれ.

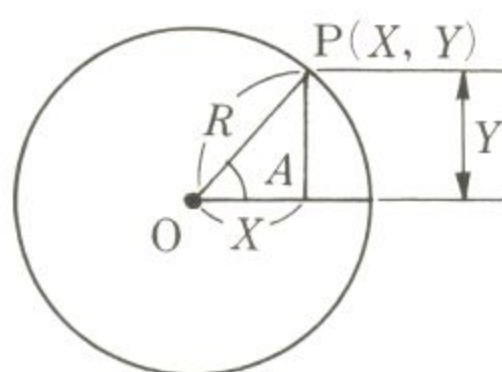
【解　説】　◎　媒介変数を使って円を表示します.

① 円の表示は媒介変数を使うと便利です. この方法は一般によく使われる方法ですから少しでもグラフィックをした人はご存知でしょう. 復習と媒介変数の意味を確認するためにあえて説明します.

② 下図の円周上の点 P は次のように表わせます.

P(X, Y)　R　A　Y　O　X

X=RCOS(A)

y=RSIN(A)

ただし, 0 を原点とする半径 R の円で X 軸と OP を結ぶ直線のつくる角を A とします.

③ このとき A の値を変えると (X, Y) の値は円周上の点 P′となります.

④ したがって, A を 0 度から 360 度まで, 細かい間隔でたとえば 1 度ずつ増やしその点を PSET (表示) するか, 1 つ後の点と順に結んでいけば円となります.

⑤ このように, A の値を変化させることで図形が描けます. この A を媒介変数と呼びます. 本章では媒介変数を使っていろいろな図形を表示します.

【プログラム例】

```
SCREEN 0
CLS 0
DEF fnx (x) = 300 + 50 * COS(x)———X の定義
DEF fny (x) = 200 - 50 * SIN(x)———Y の定義
PSET (fnx(0), fny(0))——————スタート点へポインタをおく
FOR i = 0 TO 2 * 3.14159 STEP .01——作図
    LINE -(fnx(i), fny(i))
NEXT i
END
```

【結　果】

【応用問題 13】　三ツ葉

三ツ葉の曲線をかくプログラムをつくれ.

【解　説】

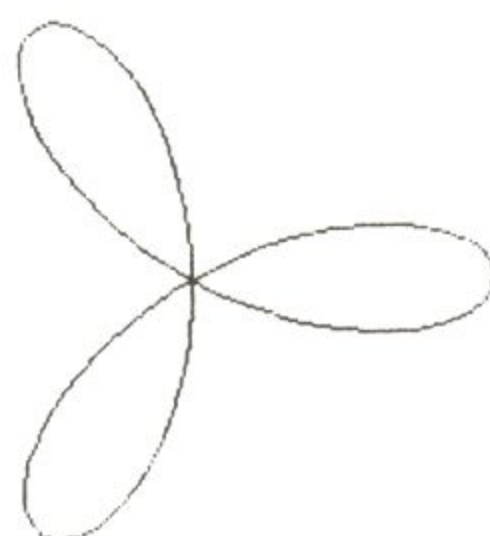
① X=300+50COS(2x)+50COS(x)

Y=200-50SIN(2x)+50SIN(x)として x を 0 ラジアンから 2π ラジアンまで 0.1 ラジアン単位で増加させて，LINE 文で結びます.

② 結果のような花びら形となります.

【プログラム例】

```
SCREEN 0
CLS 0
DEF fnx (x) = 300 + 50 * COS(2 * x) + 50 * COS(x)  ——円周上の点の X 座標値
DEF fny (x) = 200 - 50 * SIN(2 * x) + 50 * SIN(x)  ——円周上の点の Y 座標値
PSET (fnx(0), fny(0))  ——┐図の表示
FOR i = 0 TO 2 * 3.14159 + .1 STEP .1
    LINE -(fnx(i), fny(i))
NEXT i  ——————————————┘
END
```

【結　果】

【応用問題 14】　5 枚の花びらの形

【応用問題 13】の 3 枚の花びらの形を変形し，5 枚の花びらの形とするプログラムをつくれ.

【解　説】　◎　5 枚の花びらの形をつくります.

① 【応用問題 13】のプログラムを変形します.

② X=300+50COS(4x)+50COS(x)

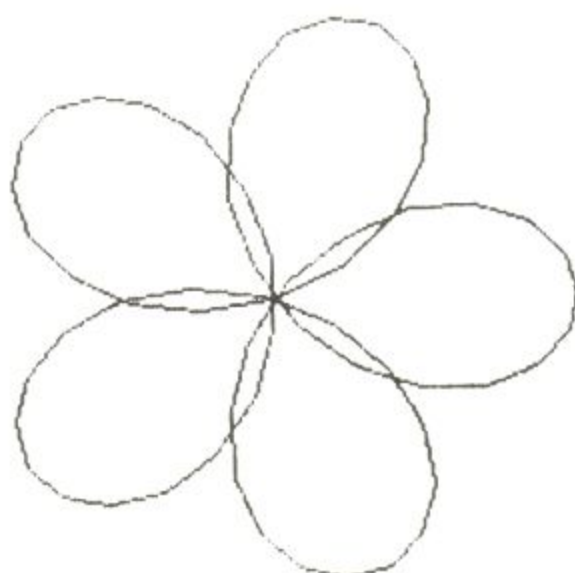
Y=200-50SIN(4x)+50SIN(x)とします.

基本形が x であった部分の一部を 4x としています.

③ この結果，5 枚の花びらの形となります. 4x の 4+1 が花びらの数です. したがって，ここを変化させれば，何枚の花びらでもかけます.

【プログラム例】

```
SCREEN 0
CLS 0
DEF fnx (x) = 300 + 50 * COS(4 * x) + 50 * COS(x) ——— X座標値
DEF fny (x) = 200 - 50 * SIN(4 * x) + 50 * SIN(x) ——— Y座標値
PSET (fnx(0), fny(0)) ——— i を 0 から 2π まで (fnx,fny) 点をむすぶ
FOR i = 0 TO 2 * 3.14159 + .1 STEP .1
    LINE -(fnx(i), fny(i))
NEXT i
END
```

【結　果】

【応用問題 15】　三ツ葉の変形 1

【応用問題 13】の花びらの形を【結　果】のように変形させるプログラムをつくれ.

【解　説】

① 【応用問題 13】のプログラムを変形します.

② X=300+50COS(2x)+50COS(x)

Y=200−100SIN(2x)+100SIN(x)とします.

基本形が 50 であった一部を 100 とします.

③ この結果, 縦方向が長くなった形となります. これは Y の値（縦方向）を変えていますから, 縦方向のみが変化するのは当然です.

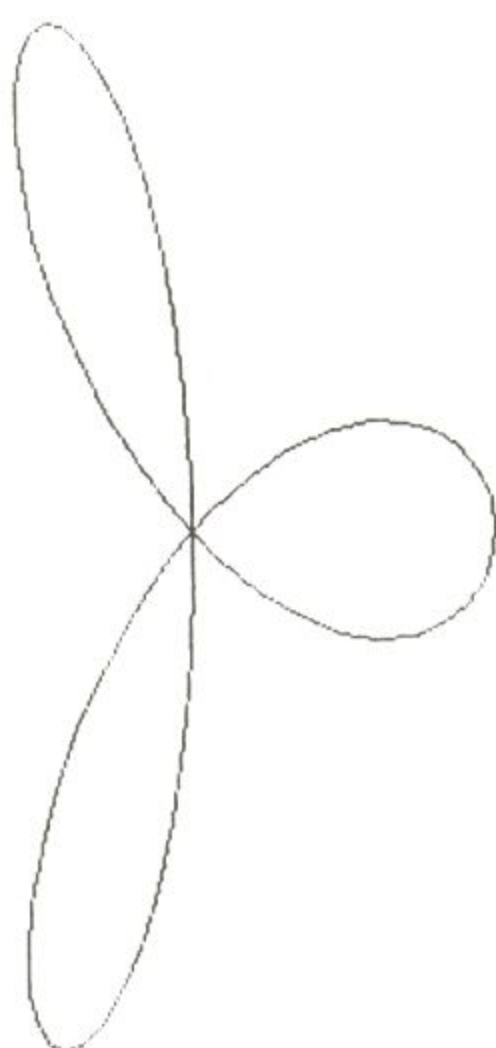
④ 100 の部分を大きくすると縦長, 逆に小さくすると縦短となります.

⑤ 簡単に類推できるように x の値を変えれば横長, 横短となります.

50COS(2x)+50COS(x)の 50 を変えて試みてください.

【プログラム例】

```
SCREEN 0
CLS 0
DEF fnx (x) = 300 + 50 * COS(2 * x) + 50 * COS(x) ——— X座標値
DEF fny (x) = 200 - 100 * SIN(2 * x) + 100 * SIN(x) ——— Y座標値
PSET (fnx(0), fny(0)) ——— i を 0 から 2π まで (fnx,fny) 点をむすぶ
FOR i = 0 TO 2 * 3.14159 + .1 STEP .1
    LINE -(fnx(i), fny(i))
NEXT i
END
```

【結　果】

【応用問題 16】　三ツ葉

【応用問題 13】の花びらの形を【結　果】のように変形させるプログラムをつくれ.

【解　説】

① 【応用問題 13】のプログラムを変形します.

② X=300+100COS(2x)+50COS(x)

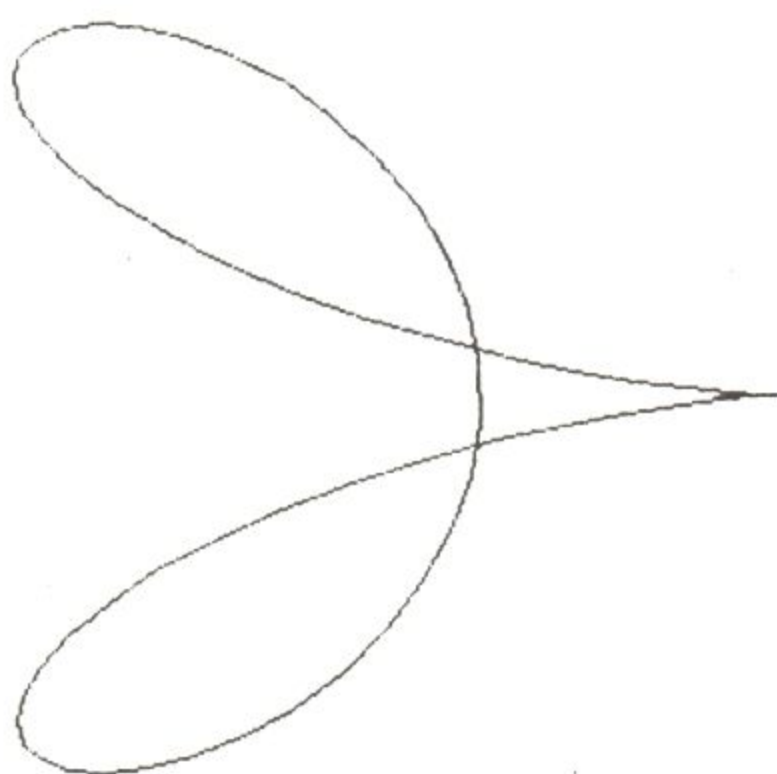
Y=200-50SIN(2x)+100SIN(x)とします.

基本形が 50 であった一部を 100 とします.

③ この結果,【結　果】のような図となります.

【プログラム例】

```
SCREEN 0
CLS 0
DEF fnx (x) = 300 + 100 * COS(2 * x) + 50 * COS(x) ——— X 座標値
DEF fny (x) = 200 - 50 * SIN(2 * x) + 100 * SIN(x) ——— Y 座標値
PSET (fnx(0), fny(0)) ——— i を 0 から 2π まで
FOR i = 0 TO 2 * 3.14159 + .1 STEP .1        (fnx,fny) 点をむすぶ
    LINE -(fnx(i), fny(i))
NEXT i
END
```

【結　果】

【応用問題 17】 無限大

無限大のマーク∞をつくるプログラムをつくれ.

【解 説】 ◎ 無限大のマークをつくります.

① X=300+50COS(x)+50COS(x)

Y=200-50SIN(COS(x)*SIN(x))とします.

② 無限大のマークとなります.

③ 4ケ所50の値, 4ケ所のxの値を変化させると無限の変形ができます. いろいろと試みてください.

【プログラム例】

```
SCREEN 0
CLS 0
DEF fnx (x) = 300 + 50 * COS(x) + 50 * COS(x)  ── X座標値
DEF fny (x) = 200 - 50 * SIN(COS(x) * SIN(x))  ── Y座標値
PSET (fnx(0), fny(0))                           ── i を 0 から 2π まで
FOR i = 0 TO 2 * 3.14159 + .1 STEP .1              (fnx,fny) 点をむすぶ
    LINE -(fnx(i), fny(i))
NEXT i
END
```

【結 果】

∞

【応用問題 18】 魚

【結 果】の魚のパターンをつくるプログラムをつくれ.

【解 説】 ◎ 魚のパターンをつくります.

① X=300+100COS(x)+100COS(COS^2(x))

Y=200-50SIN(COS(x)・SIN(x))

としてXを0から2π ラジアンまでかきます.

② この他にもいろいろのパターンがつくれますので試みてください.

【プログラム例】

```
SCREEN 0
CLS 0
DEF fnx (x) = 300 + 100 * COS(x) + 100 * COS(COS(x) * COS(x))  ── X座標値
DEF fny (x) = 200 - 50 * SIN(COS(x) * SIN(x))                  ── Y座標値
PSET (fnx(0), fny(0))                                          ── i を 0 から 2π まで
FOR i = 0 TO 2 * 3.14159 + .1 STEP .1                             (fnx,fny) 点をむすぶ
    LINE -(fnx(i), fny(i))
NEXT i
END
```

【結　果】

【応用問題 19】　アステロイド曲線

アステロイド曲線をかくプログラムをつくれ.

【解　説】　◎アステロイド曲線をかきます.

① X=300+150COS³(x)

Y=200-150SIN³(x)とします.

② 【結　果】のアステロイド曲線となります.

【プログラム例】

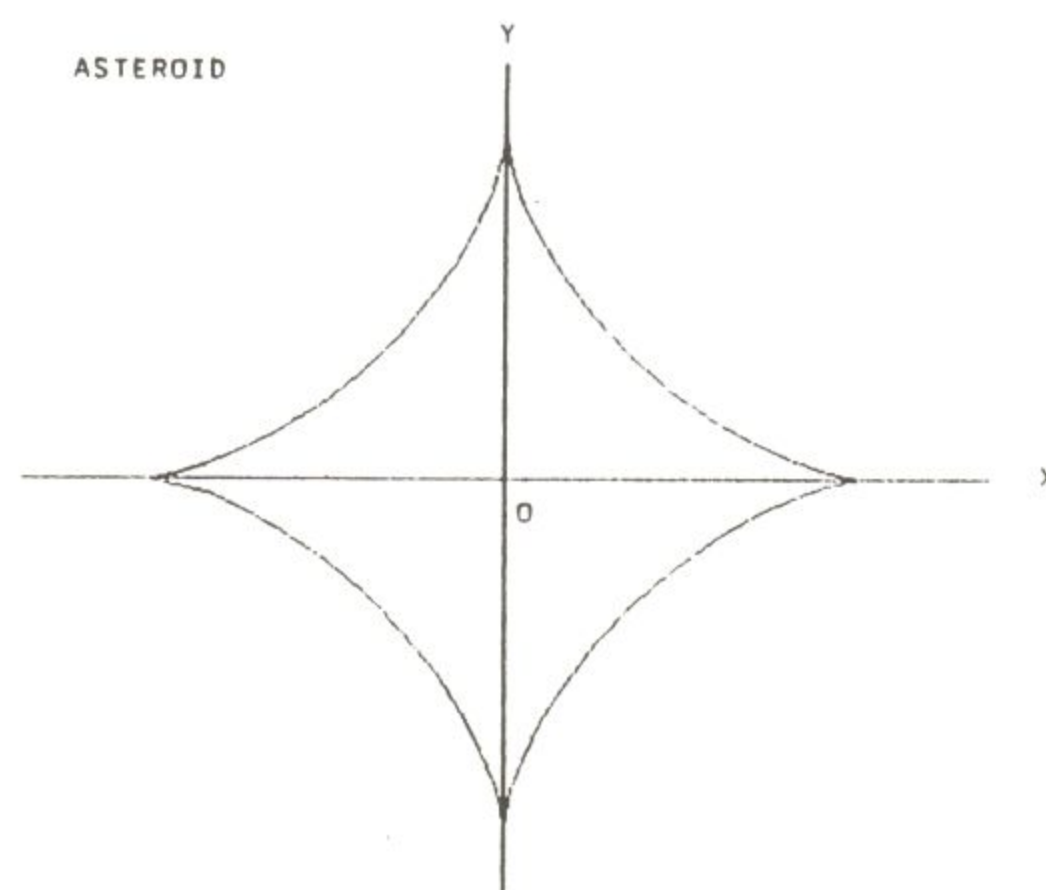
```
SCREEN 0
CLS 0
DEF fnx (x) = 300 + 150 * COS(x) * COS(x) * COS(x)
DEF fny (x) = 200 - 150 * SIN(x) * SIN(x) * SIN(x)
LOCATE 2, 16: PRINT "ASTEROID"
LINE (300, 20)-(300, 380)
LINE (100, 200)-(500, 200)
LOCATE 1, 38: PRINT "Y"
LOCATE 14, 39: PRINT "O"
LOCATE 13, 66: PRINT "X"
PSET (fnx(0), fny(0))
FOR i = 0 TO 2 * 3.14159 + .1 STEP .1
    LINE -(fnx(i), fny(i))
NEXT i
END
```

【結　果】

ASTEROID

Y

O

X

【応用問題 20】　円の伸開線（インボリュート）

円のインボリュートをかくプログラムをつくれ.

【解　説】 ◎ 円の伸開線（インボリュート）をかきます.

① インボリュートのX，Y座標は次のようになります.

X=a(cos(d)+dsin(d))

Y=a(sin(d)-dcos(d))

② X，Y座標をdを0ラジアンから40*πラジアンまで求めて結びます.

ここではaを1とします.

【プログラム例】

```
SCREEN 0
CLS 0
DEF fnx (d) = 300 + (COS(d) + d * SIN(d))────┐X, Y座標値の定義
DEF fny (d) = 200 - (SIN(d) - d * COS(d))────┘
PSET (fnx(0), fny(0))─────────────────────────┐表示
    FOR i = 0 TO 40 * 3.14159 + .1 STEP .1    │
        LINE -(fnx(i), fny(i))                │
    NEXT i ───────────────────────────────────┘
END
```

【結　果】

著 者 略 歴

田　中　　廣（たなか　ひろし）

昭和14年　東京に生れる
昭和37年　慶応義塾大学経済学部卒
昭和37年　㈱安川電機製作所入社
昭和53年　安川商事㈱入社

著　書　ビジネスプログラミング・シリーズ
マイコンプログラミング500題
パソコンプログラミング500題
「PC-8801」周辺機器活用プログラミング300題
ポケットコンピュータプログラミング300題・シリーズ
フロッピーディスク活用入門
C言語プログラミング500題
PC-9801プログラミング500題
PC-9801機械語プログラミング500題
PC-9801活用プログラミング300題
PC-9801アイデアプログラミング300題
Lotus 1-2-3活用プログラミング300題
Multiplan 3.1活用プログラミング300題
dBASE Ⅲ PLUS活用プログラミング300題
Quick Cプログラミング500題

書　名	Quick BASIC プログラミング500題　　NDC 007.64
	（定価はカバーに表示してあります）
発行年月日	1989年11月30日　初版1刷発行 1993年 7月 7日　初版5刷発行
©著　者	田　中　　廣
発 行 者	藤　吉　敏　生
発 行 所	日 刊 工 業 新 聞 社 東京都千代田区九段北一丁目8番10号 （〒102） ☎ 東京　3222-7111 振替口座　東京　9-186076
印刷・製本	大日本印刷株式会社
	1989　Printed in Japan
	乱丁・落丁本はお取り替えいたします.
	ISBN4-526-02640-9　C3034

Quick BASIC

Quick BASIC

プログラミング500題

田中　廣——著

日刊工業

定価 3200円(本体 3107円)

ISBN4-526-02640-9 C3034 P3200E